G.C.E. O/L ICT · 11 ශ්‍රේණිය

පාඩම 1: ක්‍රමලේඛනය

- 1.1 ගැටලු විශ්ලේෂණය: ආදාන, ක්‍රියාවලිය සහ ප්‍රතිදාන
- 1.2 ඇල්ගොරිතම: අනුක්‍රමය, වරණය සහ පුනරාවර්තනය
- 1.3 ගැලීම් සටහන් සහ ව්‍යාජ කේත
- 1.4 හඳුන්වන, විචල්‍ය, නියත සහ දත්ත වර්ග
- 1.5 කාරක: ගණිත, සංසන්දන සහ තාර්කික
- 1.6 වරණය: If–EndIf, If–Else–EndIf සහ Case
- 1.7 පුනරාවර්තනය: FOR, WHILE සහ REPEAT–UNTIL ලූප
- 1.8 කැදැලි පාලන ව්‍යුහ
- 1.9 ඒක-මාන අරා
- 1.10 උප වැඩසටහන්: ශ්‍රිත සහ ක්‍රියා පටිපාටි
- 1.11 ක්‍රමලේඛන භාෂා, පරිවර්තක සහ දෝෂ නිරාකරණය

© 2026 schoolict.net (Ruwan Mangala Suraweera)
සියලු හිමිකම් ඇවිරිණි.

G.C.E. O/L ICT · 11 ශ්‍රේණිය · පාඩම 1

1.1 ගැටලු විශ්ලේෂණය: ආදාන, ක්‍රියාවලිය සහ ප්‍රතිදාන

විෂය නිර්දේශය 10.1. වැඩසටහනක් ලිවීමට පෙර ගැටලුව තේරුම් ගත යුතුය. **පරිගණක වැඩසටහනක්** (program) යනු පරිගණකයට කළ යුතු දේ පියවරෙන් පියවර කියන උපදෙස් කට්ටලයකි. මෙම සටහනෙන් ගැටලුවක **ආදාන** (inputs), **ප්‍රතිදාන** (outputs) සහ **ක්‍රියාවලිය** (process) සොයා ගැනීමට, එය විසඳීමට **විවිධ ක්‍රම** ගැන සිතීමට සහ **හොඳම** ක්‍රමය තෝරා ගැනීමට ඉගෙන ගනිමු. මෙය **ගැටලු විශ්ලේෂණය** ලෙස හැඳින්වේ.

1. ක්‍රමලේඛනය යනු කුමක්ද?

පරිගණකයට තනිවම සිතිය නොහැක. එය උපදෙස් පමණක් පිළිපදී. එම උපදෙස් පරිගණකයට තේරෙන භාෂාවකින් ලිවීම **ක්‍රමලේඛනය** (programming) ලෙසත්, ඒවා ලියන්නා **ක්‍රමලේඛකයා** (programmer) ලෙසත් හැඳින්වේ. වැඩසටහනක් කුඩා විය හැක (සංඛ්‍යා දෙකක් එකතු කිරීම) හෝ ඉතා විශාල විය හැක (ශ්‍රී ලංකා දුම්රිය e-ticket පද්ධතිය, බැංකු ATM එකක් හෝ ජංගම ක්‍රීඩාවක්).

> 💡 කිරිබත් හදන වට්ටෝරුවක් ගැන සිතන්න. සහල්, පොල් කිරි සහ ලුණු **ආදාන** වේ; පිසීම **ක්‍රියාවලිය** වේ; කිරිබත් පිඟාන **ප්‍රතිදානය** වේ. වැඩසටහනක්ද එලෙසම ක්‍රියා කරයි.

2. ගැටලු විසඳීමේ පියවර

Read carefully. What is asked?
හොඳින් කියවන්න. අසා ඇත්තේ කුමක්ද?

What data do we have? What result do we want?
අප සතු දත්ත මොනවාද? අපට අවශ්‍ය ප්‍රතිඵලය කුමක්ද?

Write the steps that turn inputs into outputs.
ආදාන ප්‍රතිදාන බවට පත් කරන පියවර ලියන්න.

Compare different ways; pick the simplest and fastest.
විවිධ ක්‍රම සසඳා, සරලම සහ වේගවත්ම එක තෝරන්න.

Code it, run it and check the answers.
කේත කර, ධාවනය කර, පිළිතුරු පරීක්ෂා කරන්න.

3. ආදාන, ක්‍රියාවලිය සහ ප්‍රතිදාන (IPO)

ආදානය – වැඩසටහනට දෙන දත්ත (ටයිප් කළ, ස්කෑන් කළ, සංවේදකයකින් කියවූ). **ක්‍රියාවලිය** – දත්ත මත කරන වැඩය (ගණනය, සංසන්දනය, පිළිවෙලට සැකසීම). **ප්‍රතිදානය** – ආපසු දෙන

ප්‍රතිඵලය (තිරයේ පෙන්වීම, මුද්‍රණය, SMS ලෙස යැවීම). අප ඒවා බොහෝ විට **IPO වගුවක** ලියමු.

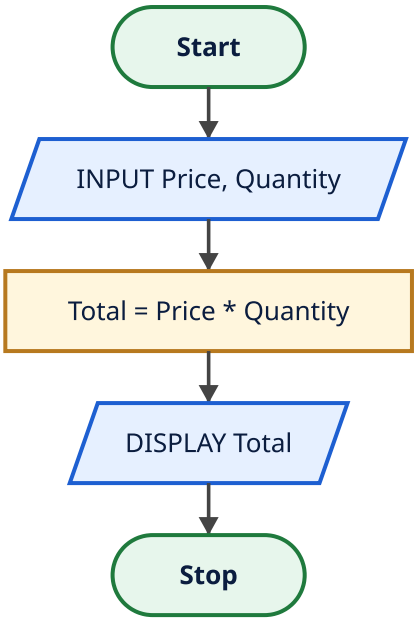
ගැටලුව	ආදානය	ක්‍රියාවලිය	ප්‍රතිදානය
කඩ බිල	මිල, ප්‍රමාණය	Total = මිල × ප්‍රමාණය	මුළු මුදල
ලකුණු 3ක සාමාන්‍යය	ලකුණු 1, 2, 3	එකතු කර 3න් බෙදන්න	සාමාන්‍යය
විදුලි බිල (CEB)	භාවිත ඒකක	ඒකක × ගාස්තුව + ස්ථාවර ගාස්තුව	ගෙවිය යුතු මුදල
සමත් හෝ අසමත්	ලකුණ	ලකුණ ≥ 35 ද?	“Pass” හෝ “Fail”
බස් ප්‍රවේශ පත්‍රය (කොළඹ–මහනුවර)	මගීන් ගණන	මගීන් × ගාස්තුව	මුළු ගාස්තුව

4. විසඳූ උදාහරණය: කඩ බිල

ගැටලුව: කඩයක සබන් එකක් රු. 150 කි. ඕනෑම සබන් ගණනක මුළු මුදල සොයන්න. **ආදානය:** මිල, ප්‍රමාණය. **ක්‍රියාවලිය:** total = මිල × ප්‍රමාණය. **ප්‍රතිදානය:** total. පහත ඇත්තේ එම විසඳුමම ව්‍යාජ කේතයක් ලෙස, ගැලීම් සටහනක් ලෙස සහ සැබෑ වැඩසටහනක් ලෙසයි. කේතය ගැන දැන් කරදර නොවන්න – 1.3 සිට 1.10 දක්වා එය පියවරෙන් පියවර ඉගෙන ගනිමු.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT Price, Quantity
  Total = Price * Quantity
  DISPLAY Total
END
```



ගැලීම් සටහන - කඩ බිල: මිල × ප්‍රමාණය

කඩ බිල: මිල × ප්‍රමාණය

Pascal

```
program ShopBill;
var
  price, qty, total: integer;
begin
  write('Enter price (Rs.): ');
  readln(price);
  write('Enter quantity: ');
  readln(qty);
  total := price * qty;
  writeln('Total = Rs. ', total);
end.
```

Python

```
price = int(input("Enter price (Rs.): "))
qty = int(input("Enter quantity: "))
total = price * qty
print("Total = Rs.", total)
```

නියැදි ධාවනය

```
Enter price (Rs.): 150
Enter quantity: 5
Total = Rs. 750
```

5. විකල්ප විසඳුම් (විසඳුම් අවකාශය)

බොහෝ ගැටලු **එක් ක්‍රමයකට වඩා** විසඳිය හැක. හැකි සියලු ක්‍රම එකට **විසඳුම් අවකාශය** (solution space) ලෙස හැඳින්වේ. උදාහරණය: 1 සිට 100 දක්වා සංඛ්‍යාවල එකතුව සොයන්න. **ක්‍රමය 1:** ලූපයක් මගින් එකින් එක එකතු කිරීම (එකතු කිරීම් 100). **ක්‍රමය 2:** N × (N + 1) ÷ 2

සූත්‍රය භාවිතය (එක් ගුණ කිරීමක් සහ එක් බෙදීමක්). දෙකම **5050** දෙයි, නමුත් ක්‍රමය 2 බෙහෙවින් වේගවත්ය.

ක්‍රමය 1: සංඛ්‍යා එකින් එක එකතු කිරීම

Pascal

```
program SumLoop;
var
  i, total: integer;
begin
  total := 0;
  for i := 1 to 100 do
    total := total + i;
  writeln('Sum = ', total);
end.
```

Python

```
total = 0
for i in range(1, 101):
    total = total + i
print("Sum =", total)
```

නියැදි ධාවනය

```
Sum = 5050
```

ක්‍රමය 2: N × (N + 1) ÷ 2 සූත්‍රය

Pascal

```
program SumFormula;
var
  n, total: integer;
begin
  n := 100;
  total := n * (n + 1) div 2;
  writeln('Sum = ', total);
end.
```

Python

```
n = 100
total = n * (n + 1) // 2
print("Sum =", total)
```

නියැදි ධාවනය

```
Sum = 5050
```

සසඳන්න	ක්‍රමය 1 (ලූපය)	ක්‍රමය 2 (සූත්‍රය)
N = 100 සඳහා පියවර ගණන	එකතු කිරීම් 100 පමණ	මෙහෙයුම් 2
තේරුම් ගැනීමට පහසුද?	ඔව්	සූත්‍රය දන්නේ නම් ඔව්
N = 1 000 000 සඳහා වේගය	මන්දගාමී	ඉතා වේගවත්

ප්‍රතිඵලය	5050	5050

6. හොඳම විසඳුම තෝරන ආකාරය

- **නිවැරදි** – සෑම වලංගු ආදානයකටම නිවැරදි පිළිතුර දෙයි.
- **කාර්යක්ෂම** – අඩු පියවර, අඩු කාලය සහ මතකය.
- **සරල** – කියවීමට, පරීක්ෂාවට සහ පසුව වෙනස් කිරීමට පහසු.
- **පිරිවැය සහ සම්පත්** – අප සතු දෘඩාංග සහ මෘදුකාංග භාවිත කරයි.

2026 නව 2026: පාසල් පරිගණකයක හෝ දුරකථනයක පවා නොමිලේ ක්‍රමලේඛනය පුහුණු විය හැක. Python සහ Pascal දෙකම **online compilers** (උදා OnlineGDB හෝ Programiz) තුළ ධාවනය වේ – ස්ථාපනය අවශ්‍ය නැත. මෙම පිටුවල වැඩසටහන් සියල්ල සැබවින්ම ධාවනය කළ අතර, නියැදි ප්‍රතිදාන සැබෑ ප්‍රතිඵලවලින් පිටපත් කළ ඒවාය.

ප්‍රධාන පද

වැඩසටහන – පරිගණකය පියවරෙන් පියවර පිළිපදින උපදෙස් කට්ටලයක්.

ක්‍රමලේඛනය – ක්‍රමලේඛන භාෂාවකින් පරිගණකයට උපදෙස් ලිවීම.

ක්‍රමලේඛකයා – වැඩසටහන් ලියන පුද්ගලයා.

ගැටලු විශ්ලේෂණය – ගැටලුවක ආදාන, ක්‍රියාවලිය සහ ප්‍රතිදාන සොයා බැලීම.

ආදානය – වැඩසටහනකට දෙන දත්ත.

ක්‍රියාවලිය – ආදාන ප්‍රතිදාන බවට හරවන වැඩය.

ප්‍රතිදානය – වැඩසටහනක් දෙන ප්‍රතිඵලය.

IPO වගුව – ගැටලුවක ආදාන, ක්‍රියාවලිය, ප්‍රතිදාන ලැයිස්තුගත කරන වගුවක්.

විසඳුම් අවකාශය – ගැටලුවක් විසඳිය හැකි සියලු ක්‍රම.

විකල්ප විසඳුම – එකම ගැටලුව විසඳන වෙනත් ක්‍රමයක්.

කාර්යක්ෂම – අඩු පියවර, අඩු කාලය සහ අඩු මතකය භාවිත කරයි.

Online compiler – කිසිවක් ස්ථාපනය නොකර වැඩසටහනක් ටයිප් කර ධාවනය කළ හැකි වෙබ් අඩවියක්.

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. වැඩසටහනක් යනු කුමක්ද?

පිළිතුර: පරිගණකයට කළ යුතු දේ පියවරෙන් පියවර කියන උපදෙස් කට්ටලයකි.

2. ගැටලු විශ්ලේෂණයේදී සොයන කොටස් තුන නම් කරන්න.

පිළිතුර: ආදානය, ක්‍රියාවලිය සහ ප්‍රතිදානය.

3. සෘජුකෝණාස්‍රයක වර්ගඵලය සොයන වැඩසටහනක ආදාන සහ ප්‍රතිදානය දෙන්න.

> **පිළිතුර:** ආදාන: දිග සහ පළල. ප්‍රතිදානය: වර්ගඵලය (ක්‍රියාවලිය: වර්ගඵලය = දිග × පළල).

4. විසඳුම් අවකාශය යනු කුමක්ද?

> **පිළිතුර:** ගැටලුවක් විසඳිය හැකි සියලු ක්‍රමවල එකතුවයි.

5. 1 සිට 100 දක්වා එකතුව සොයන ක්‍රම දෙකක් දී හොඳ එක කියන්න.

> **පිළිතුර:** ලූපයකින් එකින් එක එකතු කිරීම, හෝ N × (N + 1) ÷ 2 භාවිතය. සූත්‍රය වඩා හොඳයි – පියවර ඉතා අඩුය.

6. හොඳම විසඳුම තෝරන විට පරීක්ෂා කළ යුතු කරුණු තුනක් දෙන්න.

> **පිළිතුර:** එය නිවැරදි, කාර්යක්ෂම (වේගවත්, අඩු මතකය) සහ කියවීමට හා වෙනස් කිරීමට සරල විය යුතුය.

බහුවරණ ප්‍රශ්න

1. පරිගණකයක් පිළිපදින උපදෙස් කට්ටලයක් හඳුන්වන්නේ...

A. වැඩසටහනක්
B. ජාලයක්
C. දත්ත සමුදාය table එකක්
D. මොනිටරයක්

2. කඩ බිල් වැඩසටහනක මිලදී ගත් ප්‍රමාණය...

A. ප්‍රතිදානයකි
B. මුද්‍රකයකි
C. ආදානයකි
D. ක්‍රියාවලියකි

3. එම වැඩසටහනේම තිරයේ පෙන්වන මුළු මුදල...

A. ආදානයකි
B. විචල්‍ය නමක් පමණි
C. ප්‍රතිදානයකි
D. වෙන් කළ වචනයකි

4. “Total = මිල × ප්‍රමාණය” අයත් වන්නේ...

A. ක්‍රියාවලියට
B. ප්‍රතිදානයට
C. දෘඩාංගයට
D. ආදානයට

5. ගැටලු විසඳීමේ පළමු පියවර...

A. පරිගණකයක් මිලදී ගැනීම
B. ගැටලුව තේරුම් ගැනීම
C. වැඩසටහන ටයිප් කිරීම
D. ප්‍රතිදානය මුද්‍රණය

6. ලකුණු තුනක සාමාන්‍යය සොයන වැඩසටහනක ආදාන...

A. සාමාන්‍යය
B. මුද්‍රකය
C. 3 අංකය පමණි
D. ලකුණු තුන

7. ගැටලුවක් විසඳිය හැකි සියලු ක්‍රම එකට හඳුන්වන්නේ...

A. ප්‍රතිදාන අවකාශය
B. ආදාන උපාංගය
C. විසඳුම් අවකාශය
D. මතක අවකාශය

8. N × (N + 1) ÷ 2 භාවිතයෙන් 1 සිට 100 දක්වා එකතුව...

A. 10100
B. 5050
C. 100
D. 5000

9. 1 සිට 100 දක්වා එකින් එක එකතු කිරීමට වඩා සූත්‍ර ක්‍රමය හොඳ ඇයි?

A. එයට අවශ්‍ය පියවර ඉතා අඩුය
B. එය වෙනස් පිළිතුරක් දෙයි
C. එය වැඩි මතකයක් භාවිත කරයි
D. එයට මුද්‍රකයක් අවශ්‍යයි

10. හොඳ විසඳුමක් විය යුත්තේ...

A. දිගු සහ සංකීර්ණ
B. වෙනස් කළ නොහැකි
C. නිවැරදි, කාර්යක්ෂම සහ සරල
D. එක් ආදානයකට පමණක් නිවැරදි

11. CEB විදුලි බිල් වැඩසටහනක භාවිත කළ ඒකක ගණන...

A. ප්‍රතිදානයකි
B. විසඳුම් අවකාශයයි
C. නියත ප්‍රතිදානයකි
D. ආදානයකි

12. වැඩසටහනක ප්‍රතිඵලය පෙන්විය හැකි ප්‍රතිදාන උපාංගයක් කුමක්ද?

A. මොනිටරය
B. යතුරු පුවරුව
C. මූසිකය
D. ස්කෑනරය

13. ආදාන, ක්‍රියාවලිය සහ ප්‍රතිදාන ලැයිස්තුගත කරන වගුව...

A. සත්‍යතා වගුව
B. IPO වගුව
C. අනුරේඛන වගුව පමණි
D. ආවර්තිතා වගුව

14. මිල 150 සහ ප්‍රමාණය 5 සඳහා කඩ බිල් වැඩසටහන පෙන්වන්නේ...

A. Total = Rs. 155
B. Total = Rs. 145
C. Total = Rs. 30
D. Total = Rs. 750

15. කිසිවක් ස්ථාපනය නොකර Python හෝ Pascal වැඩසටහනක් ධාවනය කිරීමට භාවිත කළ හැක්කේ...

A. spreadsheet ප්‍රස්ථාරයක්
B. online compiler එකක්
C. වචන සැකසුම් මෘදුකාංගයක්
D. ස්කෑනරයක්

ව්‍යුහගත ප්‍රශ්න

1. (i) ගැටලු විශ්ලේෂණය යනු කුමක්ද? (ලකුණු 1)
(ii) සෘජුකෝණාස්‍රයක වර්ගඵලය සොයන වැඩසටහනක් සඳහා IPO වගුවක් අඳින්න. (ලකුණු 3) *(ලකුණු 4)*

> (i) වැඩසටහන ලිවීමට පෙර ගැටලුවක ආදාන, ක්‍රියාවලිය සහ ප්‍රතිදාන සොයා බැලීම. (ii) ආදාන: දිග, පළල (1); ක්‍රියාවලිය: වර්ගඵලය = දිග × පළල (1); ප්‍රතිදානය: වර්ගඵලය (1).

2. කොළඹ සිට මහනුවරට බසයක සෑම මගියෙකුටම එකම ගාස්තුවකි. මුළු ගාස්තුව සොයන වැඩසටහනක ආදාන, ක්‍රියාවලිය සහ ප්‍රතිදානය හඳුනාගන්න. (ලකුණු 3) *(ලකුණු 3)*

> ආදාන: මගීන් ගණන සහ ගාස්තුව (1); ක්‍රියාවලිය: total = මගීන් × ගාස්තුව (1); ප්‍රතිදානය: මුළු ගාස්තුව (1).

3. 1 සිට N දක්වා සංඛ්‍යාවල එකතුව සොයන වෙනස් ක්‍රම දෙකක් දෙන්න. වඩා කාර්යක්ෂම ක්‍රමක්ද? ඇයි? (ලකුණු 4) *(ලකුණු 4)*

> ක්‍රමය 1: ලූපයකින් 1, 2, 3 ... N එකින් එක එකතු කිරීම (1). ක්‍රමය 2: N × (N + 1) ÷ 2 භාවිතය (1). ක්‍රමය 2 වඩා කාර්යක්ෂමයි (1), මන්ද N ක්‍රමක් වුවද මෙහෙයුම් දෙකක් පමණක් භාවිත කරන අතර ලූපයට එකතු කිරීම් N ක් අවශ්‍යයි (1).

4. පරිගණකයකින් ගැටලුවක් විසඳීමේ පියවර පහ පිළිවෙළින් ලියන්න. (ලකුණු 5) *(ලකුණු 5)*

> 1 ගැටලුව තේරුම් ගැනීම; 2 ආදාන සහ ප්‍රතිදාන සොයා ගැනීම; 3 ක්‍රියාවලිය සැලසුම් කිරීම (ඇල්ගොරිතමය); 4 හොඳම විසඳුම තේරීම; 5 වැඩසටහන ලියා පරීක්ෂා කිරීම (එකට 1).

5. විසඳුම් අවකාශයෙන් හොඳම විසඳුම තෝරන විට පරීක්ෂා කළ යුතු කරුණු තුනක් දෙන්න. (ලකුණු 3) *(ලකුණු 3)*

> සියලු වලංගු ආදානවලට නිවැරදි; කාර්යක්ෂම (අඩු පියවර, කාලය/මතකය); කියවීමට, පරීක්ෂාවට, වෙනස් කිරීමට සරල; ඇති සම්පත්වලට ගැළපේ (ඕනෑම තුනක්).

සාරාංශය

- වැඩසටහන = පරිගණකයට උපදෙස් කට්ටලයක්.
- ගැටලු විශ්ලේෂණය කේත ලිවීමට පෙර සිදුවේ.
- ආදානය = ඇතුළු වන දත්ත; ප්‍රතිදානය = පිටවන ප්‍රතිඵලය.
- ක්‍රියාවලිය = ආදාන ප්‍රතිදාන කරන වැඩය.
- සැලසුමට IPO වගුවක් භාවිත කරන්න.
- විසඳුම් අවකාශය = හැකි සියලු විසඳුම්.
- හොඳම විසඳුම: නිවැරදි, කාර්යක්ෂම, සරල.
- 1–100 එකතුව: ලූපය හෝ සූත්‍රය; දෙකම 5050.

MCQ පිළිතුරු – 1.1

1. **A** – වැඩසටහන උපදෙස් කට්ටලයකි.
2. **C** – එය වැඩසටහනට දෙන දත්තයකි.
3. **C** – එය ප්‍රතිඵලයයි.
4. **A** – ගණනය = ක්‍රියාවලිය.
5. **B** – පළමුව තේරුම් ගන්න.
6. **D** – ලකුණු දෙනු ලැබේ; සාමාන්‍යය ප්‍රතිදානයයි.
7. **C** – විසඳුම් අවකාශය.
8. **B** – 100 × 101 ÷ 2 = 5050.
9. **A** – එකම පිළිතුර, අඩු පියවර.
10. **C** – නිවැරදි, කාර්යක්ෂම, සරල.
11. **D** – ඒකක මීටරයෙන් කියවා වැඩසටහනට දෙනු ලැබේ.
12. **A** – මොනිටරය ප්‍රතිදානය පෙන්වයි.
13. **B** – IPO = ආදාන-ක්‍රියාවලිය-ප්‍රතිදාන.
14. **D** – 150 × 5 = 750.

15. **B** – Online compilers browser එකේ කේතය ධාවනය කරයි.

© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.

G.C.E. O/L ICT · 11 ශ්‍රේණිය · පාඩම 1

1.2 ඇල්ගොරිතම: අනුක්‍රමය, වරණය සහ පුනරාවර්තනය

විෂය නිර්දේශය 10.2. ඇල්ගොරිතමයක් යනු ගැටලුවක් විසඳන, නිවැරදි පිළිවෙළට ඇති පැහැදිලි පියවර ලැයිස්තුවකි. තේ හදන එකේ සිට බැංකුවක් පවත්වාගෙන යාම දක්වා සෑම ඇල්ගොරිතමයක්ම **පාලන ව්‍යුහ තුනකින්** පමණක් ගොඩනැගේ: **අනුක්‍රමය** (sequence), **වරණය** (selection) සහ **පුනරාවර්තනය** (iteration). මේ තුන ඉගෙන ගත් විට ඕනෑම වැඩසටහනක් සැලසුම් කළ හැක.

1. ඇල්ගොරිතමයක් යනු කුමක්ද?

ඇල්ගොරිතමයක් යනු **පියවරෙන් පියවර සැලැස්මකි**. එය තවම වැඩසටහනක් නොවේ – එය සරල වචනවලින්, ව්‍යාජ කේතයෙන් හෝ ගැලීම් සටහනකින් ලියනු ලැබේ. හොඳ ඇල්ගොරිතමයකට: පැහැදිලි **ආරම්භයක් සහ අවසානයක්**, **පැහැදිලි** පියවර (අනුමාන නැත), **නිවැරදි පිළිවෙළ**, සහ සීමිත පියවර ගණනකින් **අවසන් වීම** තිබිය යුතුය.

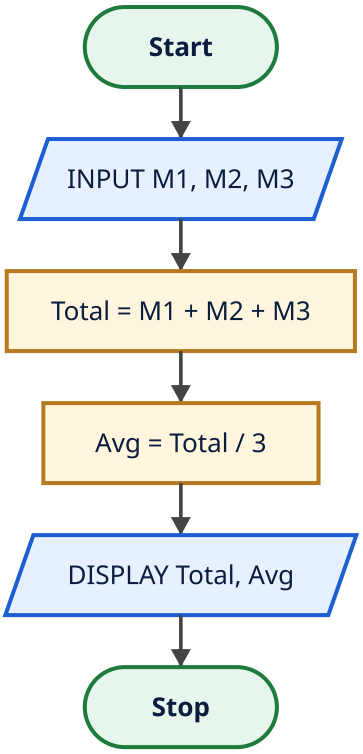
> 💡 එදිනෙදා ඇල්ගොරිතමය – තේ කෝප්පයක් හැදීම: 1 වතුර උණු කරන්න. 2 කෝප්පයට තේ බෑගයක් දමන්න. 3 උණු වතුර වත් කරන්න. 4 සීනි කැමති නම් සීනි දමන්න. 5 කලතන්න. 4 වන පියවර **වරණයකි**. “සීනි දිය වන තුරු” කලතන්නේ නම් එය **පුනරාවර්තනයකි**.

2. අනුක්‍රමය (Sequence)

අනුක්‍රමයේදී පියවර **එකින් එක**, ඉහළ සිට පහළට, සෑම පියවරක්ම හරියටම එක් වරක් ක්‍රියාත්මක වේ. උදාහරණය: ලකුණු තුනක් ආදානය කර, එකතුව සොයා, සාමාන්‍යය සොයා, ඒවා පෙන්වීම.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT M1, M2, M3
  Total = M1 + M2 + M3
  Avg = Total / 3
  DISPLAY Total, Avg
END
```



ගැලීම් සටහන - අනුක්‍රමය: ලකුණු තුනක එකතුව සහ සාමාන්‍යය

අනුක්‍රමය: ලකුණු තුනක එකතුව සහ සාමාන්‍යය

Pascal

```
program Average3;
var
  m1, m2, m3, total: integer;
  avg: real;
begin
  write('Enter mark 1: ');
  readln(m1);
  write('Enter mark 2: ');
  readln(m2);
  write('Enter mark 3: ');
  readln(m3);
  total := m1 + m2 + m3;
  avg := total / 3;
  writeln('Total = ', total);
  writeln('Average = ', avg:0:2);
end.
```

Python

```
m1 = int(input("Enter mark 1: "))
m2 = int(input("Enter mark 2: "))
m3 = int(input("Enter mark 3: "))
total = m1 + m2 + m3
avg = total / 3
print("Total =", total)
print(f"Average = {avg:.2f}")
```

නියැදි ධාවනය

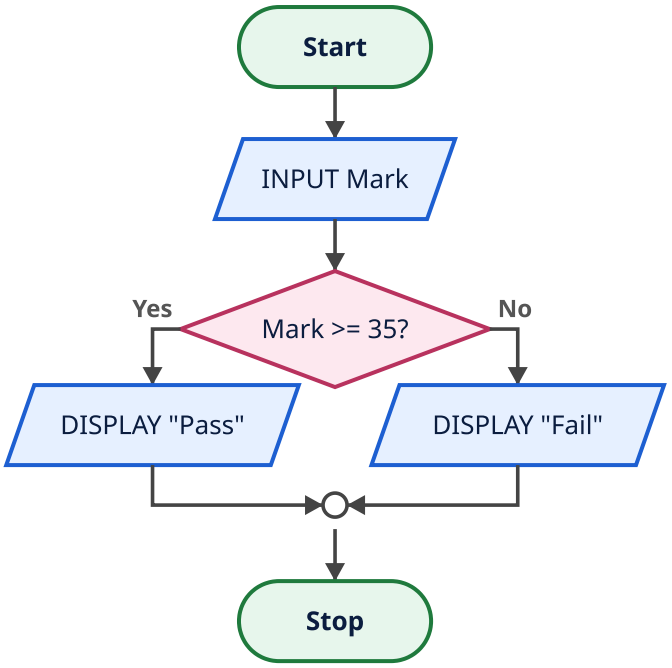
```
Enter mark 1: 75
Enter mark 2: 68
Enter mark 3: 72
Total = 215
Average = 71.67
```

3. වරණය (Selection)

වරණයේදී පරිගණකය **කොන්දේසියක්** (ඔව්/නැත පිළිතුරක් ඇති ප්‍රශ්නයක්) පරීක්ෂා කර ක්‍රියාත්මක කළ යුතු පියවර **තෝරා ගනී**. උදාහරණය: ලකුණ 35 හෝ ඊට වැඩි නම් “Pass” ද, නැතිනම් “Fail” ද පෙන්වන්න. වරණය **තීරණයක්** (decision) ලෙසද හැඳින්වේ.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT Mark
  IF Mark >= 35 THEN
    DISPLAY "Pass"
  ELSE
    DISPLAY "Fail"
  ENDIF
END
```



ගැලීම් සටහන – වරණය: සමත් හෝ අසමත්

වරණය: සමත් හෝ අසමත්

Pascal

```
program PassFail;
var
  mark: integer;
begin
  write('Enter mark: ');
  readln(mark);
  if mark >= 35 then
    writeln('Pass')
  else
    writeln('Fail');
end.
```

Python

```
mark = int(input("Enter mark: "))
if mark >= 35:
    print("Pass")
else:
    print("Fail")
```

නියැදි ධාවනය 1

```
Enter mark: 52
Pass
```

නියැදි ධාවනය 2

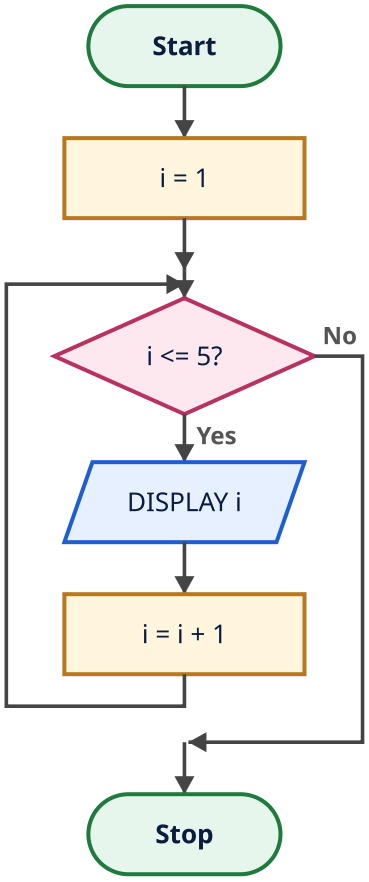
```
Enter mark: 28
Fail
```

4. පුනරාවර්තනය (Iteration)

පුනරාවර්තනයේදී (**පුනරාවර්තනය** හෝ **ලූපය** ලෙසද හැඳින්වේ) කොන්දේසියක් සත්‍ය වන තාක් හෝ නියමිත වාර ගණනක් සමහර පියවර **නැවත නැවත** ක්‍රියාත්මක වේ. උදාහරණය: 1 සිට 5 දක්වා සංඛ්‍යා පෙන්වීම. ලූපයක් නැතිනම් DISPLAY පියවර පහක් අවශ්‍යයි; ලූපයක් සමඟ එකක් ප්‍රමාණවත්ය.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  FOR i = 1 TO 5
    DISPLAY i
  ENDFOR
END
```



ගැලීම් සටහන – පුනරාවර්තනය: 1 සිට 5 දක්වා පෙන්වීම

පුනරාවර්තනය: 1 සිට 5 දක්වා පෙන්වීම

Pascal

```
program Count5;
var
  i: integer;
begin
  for i := 1 to 5 do
    writeln(i);
end.
```

Python

```
for i in range(1, 6):
    print(i)
```

නියැදි ධාවනය

```
1
2
3
4
5
```

5. ව්‍යුහ තුන එක ළඟ

ව්‍යුහය	සිදුවන දේ	ව්‍යාජ කේත වචන	එදිනෙදා උදාහරණය
අනුක්‍රමය	පියවර පිළිවෙළින්, එක් වරක් බැගින්	INPUT, පැවරීම, DISPLAY	අවදි වීම, මුහුණ සේදීම, උදෑසන ආහාරය
වරණය	කොන්දේසියක් මාර්ගය තෝරයි	IF … THEN … ELSE … ENDIF	වැසි නම් කුඩය ගන්න
පුනරාවර්තනය	පියවර නැවත නැවත	FOR, WHILE, REPEAT … UNTIL	පාසල් ක්‍රීඩාංගණය වටා වට 4ක් දිවීම

2026 නව 2026 – **Scratch සහ block coding.** Scratch (scratch.mit.edu) තුළ කුට්ටි එකට සම්බන්ධ කරමින් වැඩසටහන් සාදයි. එම ව්‍යුහ තුනම එහි ඇත: එකක් යට එකක් තැබූ කුට්ටි **අනුක්‍රමයකි**, “if … then … else” කුට්ටිය **වරණයයි**, “repeat 10” හෝ “repeat until” **පුනරාවර්තනයයි**. Pascal හෝ Python ටයිප් කිරීමට පෙර ඇල්ගොරිතම පුහුණුවට Scratch ඉතා හොඳය.

ප්‍රධාන පද

ඇල්ගොරිතමය – ගැටලුවක් විසඳන පැහැදිලි, පිළිවෙළට ඇති පියවර ලැයිස්තුවක්.

පාලන ව්‍යුහය – පියවර ක්‍රියාත්මක වන පිළිවෙළ පාලනය කරන ක්‍රමයක්.

අනුක්‍රමය – පියවර එකින් එක, එක් වරක් බැගින් ක්‍රියාත්මක වීම.

වරණය – කොන්දේසියක් මගින් ක්‍රියාත්මක කළ යුතු පියවර තේරීම.

පුනරාවර්තනය – පියවර කණ්ඩායමක් නැවත නැවත ක්‍රියාත්මක කිරීම.	**ලූපය** – පුනරාවර්තනයට තවත් නමක්.
කොන්දේසිය – පිළිතුර සත්‍ය (ඔව්) හෝ අසත්‍ය (නැත) වන ප්‍රශ්නයක්.	**තීරණය** – වරණයට තවත් නමක්.
සීමිත – ඇල්ගොරිතමයක් සීමිත පියවර ගණනකින් අවසන් විය යුතුය.	**සාමාන්‍යය** – එකතුව අගයන් ගණනින් බෙදීම.
Scratch – නොමිලේ ලැබෙන කුට්ටි මත පදනම් වූ ක්‍රමලේඛන මෙවලමක්.	**Block coding** – කේත ටයිප් කිරීම වෙනුවට කුට්ටි සම්බන්ධ කර ක්‍රමලේඛනය.

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. ඇල්ගොරිතමයක් යනු කුමක්ද?

> **පිළිතුර:** ගැටලුවක් විසඳා අවසන් වන, නිවැරදි පිළිවෙළට ඇති පැහැදිලි පියවර ලැයිස්තුවකි.

2. පාලන ව්‍යුහ තුන නම් කරන්න.

> **පිළිතුර:** අනුක්‍රමය, වරණය සහ පුනරාවර්තනය.

3. Pass හෝ Fail තීරණය කිරීමට භාවිත කරන ව්‍යුහය කුමක්ද?

> **පිළිතුර:** වරණය.

4. 1 සිට 100 දක්වා පෙන්වීමට භාවිත කරන ව්‍යුහය කුමක්ද?

> **පිළිතුර:** පුනරාවර්තනය (ලූපයක්).

5. ලකුණු 75, 68 සහ 72 සඳහා එකතුව සහ සාමාන්‍යය කුමක්ද?

> **පිළිතුර:** එකතුව = 215, සාමාන්‍යය = 71.67.

6. හොඳ ඇල්ගොරිතමයක ගුණාංග දෙකක් දෙන්න.

> **පිළිතුර:** පැහැදිලි පියවර; නිවැරදි පිළිවෙළ; පැහැදිලි ආරම්භය සහ අවසානය; අවසන් වීම (ඕනෑම දෙකක්).

බහුවරණ ප්‍රශ්න

1. ගැටලුවක් විසඳන පැහැදිලි, පිළිවෙළට ඇති පියවර ලැයිස්තුවක්...

A. මෙහෙයුම් පද්ධතියකි
B. ඇල්ගොරිතමයකි
C. සම්පාදකයකි
D. දත්ත සමුදායකි

2. එකින් එක, එක් වරක් බැගින් ක්‍රියාත්මක වන පියවර...

A. පුනරාවර්තනයකි
B. වරණයකි
C. ලූපයකි
D. අනුක්‍රමයකි

3. “වැසි නම් කුඩය ගන්න” යනු ... සඳහා උදාහරණයකි

A. වරණය
B. පුනරාවර්තනය
C. අනුක්‍රමය
D. අරාව

4. “ක්‍රීඩාංගණය වටා වට 4ක් දුවන්න” යනු ... සඳහා උදාහරණයකි

A. අනුක්‍රමය පමණි
B. පුනරාවර්තනය
C. නියතයක්
D. වරණය

5. පාලන ව්‍යුහ තුනෙන් එකක් නොවන්නේ කුමක්ද?

A. සම්පාදනය
B. අනුක්‍රමය
C. වරණය
D. පුනරාවර්තනය

6. කොන්දේසියක පිළිතුර සැමවිටම...

A. නමක්
B. පූර්ණ සංඛ්‍යාවක්
C. සත්‍ය හෝ අසත්‍ය
D. පින්තූරයක්

7. ලකුණු 75, 68 සහ 72 හි එකතුව...

A. 215
B. 205
C. 71.67
D. 225

8. ලකුණ 28 සඳහා Pass/Fail වැඩසටහන පෙන්වන්නේ...

A. 28
B. Pass
C. Nothing
D. Fail

9. පුනරාවර්තනයට තවත් නමක්...

A. නියතය
B. ලූපය
C. තීරණය
D. ප්‍රතිදානය

10. වරණයට තවත් නමක්...

A. තීරණය
B. ලූපය
C. විචල්‍යය
D. අනුක්‍රමය

11. ඇල්ගොරිතමයක් සීමිත පියවර ගණනකින් අවසන් විය යුතුය. මෙයින් අදහස් වන්නේ එය...

A. අසීමිත බවයි
B. සැඟවුණු බවයි
C. සීමිත බවයි
D. අහඹු බවයි

12. count5 වැඩසටහන සංඛ්‍යා කීයක් පෙන්වයි?

A. 4
B. 1
C. 5
D. 6

13. Scratch හි “repeat 10” කුට්ටිය...

A. ප්‍රතිදාන උපාංගයකි
B. පුනරාවර්තනයයි
C. විචල්‍යයකි
D. වරණයයි

14. Scratch හි “if ... then ... else” කුට්ටිය...

A. අරාවකි
B. අනුක්‍රමයයි
C. පුනරාවර්තනයයි
D. වරණයයි

15. වැඩසටහන සැලසුම් කිරීමට එයට පෙර ලියන්නේ මින් කුමක්ද?

A. ප්‍රතිදානය
B. සම්පාදකය
C. ඇල්ගොරිතමයක්
D. දෝෂ පණිවිඩය

ව්‍යුහගත ප්‍රශ්න

1. (i) ඇල්ගොරිතමයක් අර්ථ දක්වන්න. (ලකුණු 1)
(ii) පාලන ව්‍යුහ තුන නම් කර එක් එකකට එදිනෙදා උදාහරණයක් දෙන්න. (ලකුණු 6) *(ලකුණු 7)*

> (i) ගැටලුවක් විසඳන පැහැදිලි, පිළිවෙළට ඇති, සීමිත පියවර ලැයිස්තුවක්. (ii) අනුක්‍රමය – පාසලට සූදානම් වීම පියවරෙන් පියවර; වරණය – වැසි නම් කුඩය ගැනීම; පුනරාවර්තනය – වට 4ක් දිවීම (නමට 1 + උදාහරණයට 1).

2. සෘජුකෝණාස්‍රයක වර්ගඵලය සොයන ඇල්ගොරිතමයක් සරල පියවරෙන් ලියන්න. එය භාවිත කරන පාලන ව්‍යුහය කුමක්ද? (ලකුණු 4) *(ලකුණු 4)*

> 1 ආරම්භය. 2 දිග සහ පළල ආදානය. 3 වර්ගඵලය = දිග × පළල. 4 වර්ගඵලය පෙන්වීම. 5 නැවතීම (3). එය අනුක්‍රමය භාවිත කරයි (1).

3. ලකුණ 35 හෝ වැඩි නම් “Pass” සහ නැතිනම් “Fail” පෙන්වන පියවර ලියන්න. මෙය කුමන ව්‍යුහයද? (ලකුණු 3) *(ලකුණු 3)*

> ලකුණ ආදානය; ලකුණ ≥ 35 නම් “Pass” නැතිනම් “Fail” පෙන්වීම (2). වරණය (1).

4. 1 සිට 100 දක්වා පෙන්වීමට DISPLAY 100 වරක් ලිවීමට වඩා ලූපයක් හොඳ ඇයි? (ලකුණු 2) *(ලකුණු 2)*

> ලූපයට පේළි කිහිපයක් පමණක් අවශ්‍යයි (1), වෙනස් කිරීම පහසුය (උදා 1–1000 ට) සහ වැරදි අඩුය (1).

5. තේ කෝප්පයක් හදන පියවර ලියන්න. වරණයක් වන එක් පියවරක් සහ පුනරාවර්තනයක් විය හැකි එක් පියවරක් සලකුණු කරන්න. (ලකුණු 4) *(ලකුණු 4)*

> වතුර උණු කිරීම; තේ බෑගය කෝප්පයට; වතුර වත් කිරීම; සීනි අවශ්‍ය නම් සීනි දැමීම (වරණය); සීනි දිය වන තුරු කලතීම (පුනරාවර්තනය) (පියවරට 2, සලකුණු කිරීමට 1 + 1).

සාරාංශය

- ඇල්ගොරිතමය = පැහැදිලි, පිළිවෙළට, සීමිත පියවර.
- පාලන ව්‍යුහ තුන: අනුක්‍රමය, වරණය, පුනරාවර්තනය.
- අනුක්‍රමය: පියවර එකින් එක.
- වරණය: කොන්දේසියක් මාර්ගය තෝරයි.
- පුනරාවර්තනය: පියවර නැවත නැවත (ලූපය).
- කොන්දේසියක් සත්‍ය හෝ අසත්‍ය වේ.
- කේත කිරීමට පෙර ඇල්ගොරිතමය සැලසුම් කරන්න.
- Scratch කුට්ටිද එම ව්‍යුහ තුනම භාවිත කරයි.

MCQ පිළිතුරු – 1.2

1. **B** – ඇල්ගොරිතමයේ අර්ථය.
2. **D** – අනුක්‍රමය.
3. **A** – කොන්දේසියක් තෝරයි.
4. **B** – 4 වරක් පුනරාවර්තනය.
5. **A** – සම්පාදනය කේත පරිවර්තනයයි.
6. **C** – ඔව්/නැත = සත්‍ය/අසත්‍ය.
7. **A** – 75+68+72 = 215.
8. **D** – 28 < 35.
9. **B** – ලූපය / පුනරාවර්තනය.
10. **A** – තීරණය.
11. **C** – සීමිත.
12. **C** – 1, 2, 3, 4, 5.
13. **B** – එය පුනරාවර්තනය කරයි.
14. **D** – එය තෝරයි.
15. **C** – පළමුව සැලසුම.

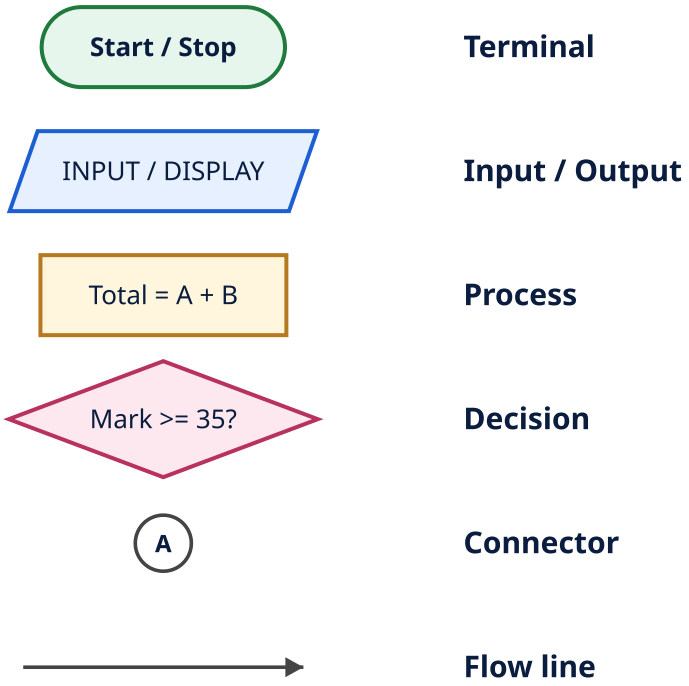
© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.

G.C.E. O/L ICT · 11 ශ්‍රේණිය · පාඩම 1

1.3 ගැලීම් සටහන් සහ ව්‍යාජ කේත

විෂය නිර්දේශය 10.3. ඇල්ගොරිතමයක් ප්‍රධාන ක්‍රම දෙකකින් පෙන්වමු. **ගැලීම් සටහනක්** (flowchart) යනු ඊතලවලින් සම්බන්ධ කළ සම්මත සංකේතවලින් සෑදූ **රූපයකි**. **ව්‍යාජ කේතය** (pseudocode) යනු සැබෑ වැඩසටහනකට සමීප, **කෙටි ඉංග්‍රීසි වැනි පේළිවලින්** ලියූ ඇල්ගොරිතමයයි. මෙම සටහනෙන් O/L ගැලීම් සටහන් සංකේත, ව්‍යාජ කේත වචන, **එකක් අනෙකට පරිවර්තනය** කරන ආකාරය සහ **අනුරේඛන වගුවකින්** (trace table) ඇල්ගොරිතමයක් පරීක්ෂා කරන ආකාරය ඉගෙන ගනිමු.

1. ගැලීම් සටහන් සංකේත



ශ්‍රී ලංකා O/L ප්‍රශ්න පත්‍රවල භාවිත සම්මත ගැලීම් සටහන් සංකේත

සංකේතය	නම	භාවිතය
වටකුරු සෘජුකෝණාස්‍රය (ඕවලය)	ආරම්භ/අවසාන	ආරම්භය සහ නැවතීම
සමාන්තරාස්‍රය	ආදාන / ප්‍රතිදාන	INPUT, DISPLAY, PRINT
සෘජුකෝණාස්‍රය	ක්‍රියාවලිය	ගණනය කිරීම්, Total = A + B
දියමන්තිය (රොම්බසය)	තීරණය	ඔව්/නැත ප්‍රශ්නයක්, උදා Mark >= 35?
කුඩා රවුම	සම්බන්ධකය	ගැලීම් රේඛා සම්බන්ධ කිරීම
ඊතලය	ගැලීම් රේඛාව	පියවරවල දිශාව පෙන්වයි

> 💡 පිළිවෙළ ගැලීම් සටහනක නීති: එක් ආරම්භයක් සහ එක් නැවතීමක්; ඊතල ඉහළ සිට පහළට; තීරණ දියමන්තියකට **ඔව්** සහ **නැත** ලෙස ලේබල් කළ හරියටම පිටවීම් **දෙකක්**; එක් එක් කොටුව තුළ කෙටි පාඨයක් ලියන්න.

2. ව්‍යාජ කේත වචන

ව්‍යාජ කේතය සැබෑ භාෂාවක් නොවන නිසා කුඩා වෙනස්කම් පිළිගැනේ. නමුත් O/L පිළිතුරුවල සාමාන්‍යයෙන් ලොකු අකුරින් ලියන මෙම වචන භාවිත කරමු:

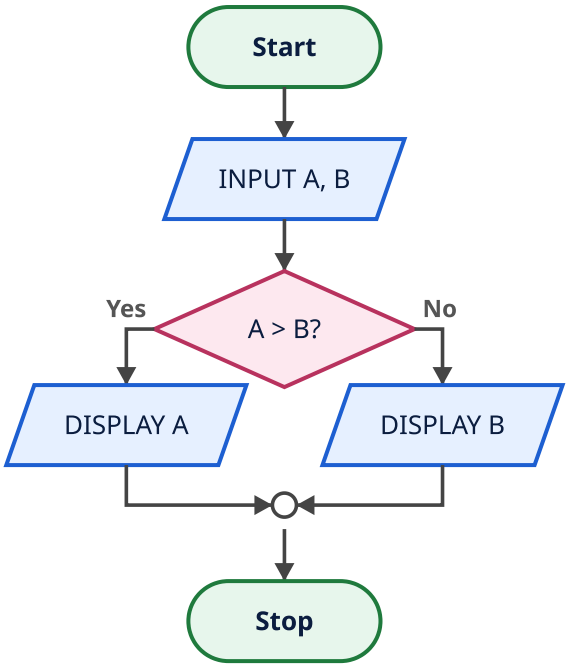
වචනය	තේරුම	ගැලීම් සටහන් සංකේතය
BEGIN / END	ඇල්ගොරිතමයේ ආරම්භය සහ අවසානය	ආරම්භ/අවසාන
INPUT / READ	දත්ත ලබා ගැනීම	ආදාන/ප්‍රතිදාන
DISPLAY / PRINT	ප්‍රතිඵලයක් පෙන්වීම	ආදාන/ප්‍රතිදාන
Total = A + B	ගණනය කර ගබඩා කිරීම	ක්‍රියාවලිය
IF … THEN … ELSE … ENDIF	වරණය	තීරණය
FOR … ENDFOR, WHILE … ENDWHILE, REPEAT … UNTIL	පුනරාවර්තනය	තීරණය + ආපසු යන ඊතලය

3. උදාහරණය: සංඛ්‍යා දෙකකින් විශාල එක

පහත ආකාර තුන බලන්න. සෑම ගැලීම් සටහන් කොටුවක්ම ව්‍යාජ කේත පේළියකට ද, සෑම ව්‍යාජ කේත පේළියක්ම කේත පේළි එකකට හෝ දෙකකට ද ගැළපේ.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT A, B
  IF A > B THEN
    DISPLAY A
  ELSE
    DISPLAY B
  ENDIF
END
```



ගැලීම් සටහන - සංඛ්‍යා දෙකකින් විශාලතම

සංඛ්‍යා දෙකකින් විශාලතම

Pascal

```
program Largest2;
var
  a, b: integer;
begin
  write('Enter first number: ');
  readln(a);
  write('Enter second number: ');
  readln(b);
  if a > b then
    writeln('Largest = ', a)
  else
    writeln('Largest = ', b);
end.
```

Python

```
a = int(input("Enter first number: "))
b = int(input("Enter second number: "))
if a > b:
    print("Largest =", a)
else:
    print("Largest =", b)
```

නියැදි ධාවනය 1

```
Enter first number: 45
Enter second number: 72
Largest = 72
```

නියැදි ධාවනය 2

```
Enter first number: 90
Enter second number: 34
Largest = 90
```

4. ගැලීම් සටහනක් ව්‍යාජ කේතයට පරිවර්තනය

Start becomes BEGIN, Stop becomes END.
Start → BEGIN, Stop → END.

Copy the text inside.
ඇතුළත පාඨය පිටපත් කරන්න.

e.g. Sum = Sum + i
උදා Sum = Sum + i

Yes side after THEN, No side after ELSE, then ENDIF.
ඔව් පැත්ත THEN ට පසුව, නැත පැත්ත ELSE ට පසුව, පසුව ENDIF.

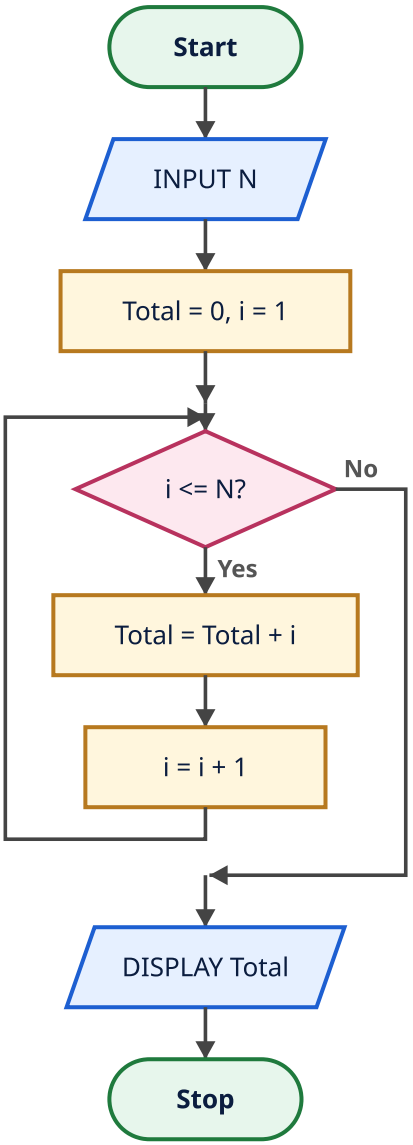
Steps inside the loop go between WHILE and ENDWHILE.
ලූපය තුළ පියවර WHILE සහ ENDWHILE අතර.

ව්‍යාජ කේතය ගැලීම් සටහනකට පරිවර්තනය කිරීම එම වගුවම පසුපසට කියවීමයි: සෑම පේළියකටම එක් සංකේතයක් අඳින්න, සෑම ලූපයකටම දියමන්තියට ආපසු ඊතලයක් අඳින්න.

5. ලූපයක් සහිත උදාහරණය: 1 සිට N දක්වා එකතුව

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT N
  Total = 0
  i = 1
  WHILE i <= N DO
    Total = Total + i
    i = i + 1
  ENDWHILE
  DISPLAY Total
END
```



ගැලීම් සටහන – 1 සිට N දක්වා සංඛ්‍යාවල එකතුව

1 සිට N දක්වා සංඛ්‍යාවල එකතුව

Pascal

```
program SumToN;
var
  n, i, total: integer;
begin
  write('Enter N: ');
  readln(n);
  total := 0;
  i := 1;
  while i <= n do
  begin
    total := total + i;
    i := i + 1;
  end;
  writeln('Sum = ', total);
end.
```

Python

```
n = int(input("Enter N: "))
total = 0
i = 1
while i <= n:
    total = total + i
    i = i + 1
print("Sum =", total)
```

නියැදි ධාවනය

```
Enter N: 5
Sum = 15
```

6. අනුරේඛන වගු (dry run)

අනුරේඛන වගුවක් සෑම පියවරකටම පසු එක් එක් විචල්‍යයේ අගය පෙන්වයි. මෙය කඩදාසියේ කිරීම **dry run** ලෙස හැඳින්වේ. වැඩසටහන ධාවනයට පෙර වැරදි සොයා ගැනීමට එය උපකාරී වේ. N = 5 සඳහා ඉහත වැඩසටහනේ අනුරේඛනය මෙන්න:

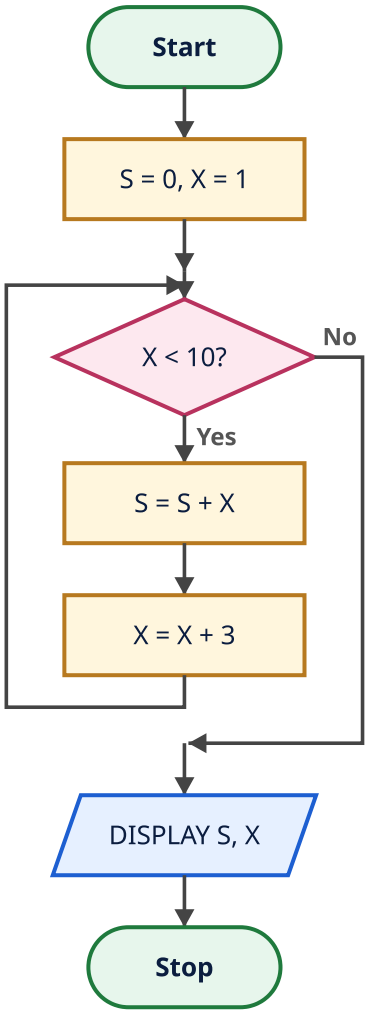
පියවර	i	Total	i <= 5 ?
start	1	0	–
loop	1	1	ඔව්
loop	2	3	ඔව්
loop	3	6	ඔව්
loop	4	10	ඔව්
loop	5	15	ඔව්
check	6	15	නැත

අවසාන පේළියේ i = 6 නිසා i <= 5 කොන්දේසිය නැත වී ලූපය නවතී. Total = **15**, එය සැබෑ ප්‍රතිදානය වන “Sum = 15” ට ගැළපේ.

7. උත්සාහ කරන්න: මෙම ලූපය අනුරේඛනය කරන්න

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  S = 0
  X = 1
  WHILE X < 10 DO
    S = S + X
    X = X + 3
  ENDWHILE
  DISPLAY S, X
END
```



ගැලීම් සටහන – මෙම loop එක trace කරන්න

පියවර	S	X	X < 10 ?
start	0	1	–
loop	1	4	ඔව්

loop	5	7	ඔව්
loop	12	10	ඔව්
check	12	10	නැත

ප්‍රතිදානය: **12 10** (S = 12, X = 10). පරීක්ෂාව: 1 + 4 + 7 = 12.

ප්‍රධාන පද

ගැලීම් සටහන – සම්මත සංකේත සහ ඊතල භාවිතයෙන් ඇල්ගොරිතමයක රූප සටහනක්.

ව්‍යාජ කේතය – කෙටි ඉංග්‍රීසි වැනි පේළිවලින් ලියූ ඇල්ගොරිතමයක්.

ආරම්භ/අවසාන සංකේතය – Start සහ Stop සඳහා වටකුරු සෘජුකෝණාස්‍රය.

ආදාන/ප්‍රතිදාන සංකේතය – INPUT සහ DISPLAY සඳහා සමාන්තරාස්‍රය.

ක්‍රියාවලි සංකේතය – ගණනය හෝ පැවරීමක් සඳහා සෘජුකෝණාස්‍රය.

තීරණ සංකේතය – ඔව්/නැත ප්‍රශ්නයක් සඳහා දියමන්තිය.

සම්බන්ධකය – ගැලීම් රේඛා සම්බන්ධ කරන කුඩා රවුම.

ගැලීම් රේඛාව – පියවරවල පිළිවෙළ පෙන්වන ඊතලය.

අනුරේඛන වගුව – සෑම පියවරකටම පසු විචල්‍ය අගයන් පෙන්වන වගුවක්.

Dry run – කඩදාසියේ අතින් ඇල්ගොරිතමයක් අනුගමනය කිරීම.

පැවරීම – විචල්‍යයක අගයක් ගබඩා කිරීම, උදා Total = 0.

පරිවර්තනය – ගැලීම් සටහනක් ව්‍යාජ කේතයට හෝ අනෙක් පසට වෙනස් කිරීම.

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. තීරණයක් සඳහා භාවිත කරන සංකේතය කුමක්ද?

> **පිළිතුර:** ඔව් සහ නැත පිටවීම් ඇති දියමන්තිය.

2. INPUT සහ DISPLAY සඳහා භාවිත කරන සංකේතය කුමක්ද?

> **පිළිතුර:** සමාන්තරාස්‍රය.

3. ව්‍යාජ කේතය යනු කුමක්ද?

> **පිළිතුර:** වැඩසටහනකට සමීප කෙටි ඉංග්‍රීසි වැනි පේළිවලින් ලියූ ඇල්ගොරිතමයකි.

4. දියමන්තියක් ලූපයක කොටසක් බව දැනගන්නේ කෙසේද?

> **පිළිතුර:** පහළ සිට එයට ආපසු ඉහළට ඊතලයක් පැමිණේ.

5. අනුරේඛන වගුවක් භාවිත කරන්නේ කුමකටද?

> **පිළිතුර:** විචල්‍ය අගයන් පියවරෙන් පියවර සටහන් කර වැරදි සොයා ගැනීමට (dry run).

6. S = 0, X = 1; WHILE X < 10: S = S + X, X = X + 3 අනුරේඛනය කරන්න. අවසාන S සහ X දෙන්න.

> **පිළිතුර:** S = 12, X = 10.

බහුවරණ ප්‍රශ්න

1. Start සහ Stop සඳහා ගැලීම් සටහන් සංකේතය...

A. වටකුරු සෘජුකෝණාස්‍රය (terminal)
B. සමාන්තරාස්‍රය
C. දියමන්තිය
D. කුඩා රවුම

2. “Total = A + B” අඳින්නේ...

A. දියමන්තියක
B. සමාන්තරාස්‍රයක
C. සෘජුකෝණාස්‍රයක (process)
D. terminal එකක

3. “DISPLAY Total” අඳින්නේ...

A. සමාන්තරාස්‍රයක
B. දියමන්තියක
C. රවුමක
D. සෘජුකෝණාස්‍රයක

4. තීරණ දියමන්තියකට පිටවීම් කීයක් තිබේද?

A. 3
B. 2
C. 1
D. 4

5. ගැලීම් සටහනක කුඩා රවුමක්...

A. terminal එකකි
B. ක්‍රියාවලියකි
C. සම්බන්ධකයකි
D. තීරණයකි

6. ව්‍යාජ කේතය...

A. ගැලීම් සටහන් සංකේත වර්ගයකි
B. පරිගණකය ධාවනය කරන ක්‍රමලේඛන භාෂාවකි
C. යන්ත්‍ර කේතයකි
D. ඉංග්‍රීසි වැනි පේළිවලින් ඇල්ගොරිතමයකි

7. ආපසු ඊතලයක් පැමිණෙන දියමන්තියක් පෙන්වන්නේ...

A. අවසානය
B. අනුක්‍රමයක්
C. ලූපයක්
D. ආදානයක්

8. කඩදාසියේ අතින් ඇල්ගොරිතමයක් අනුගමනය කිරීම හඳුන්වන්නේ...

A. upload
B. download
C. compile
D. dry run

9. N = 5 සමඟ 1 සිට N දක්වා එකතු කරන වැඩසටහනේ ප්‍රතිදානය...

A. Sum = 5
B. Sum = 15
C. Sum = 10
D. Sum = 21

10. එම වැඩසටහනේ ලූපය නවතින විට i හි අගය...

A. 6
B. 15
C. 5
D. 0

11. 45 සහ 72 ආදාන සඳහා විශාල සංඛ්‍යා වැඩසටහන පෙන්වන්නේ...

A. Largest = 45
B. Largest = 72
C. Largest = 27
D. Largest = 117

12. අනුරේඛනය: S = 0, X = 1; WHILE X < 10: S = S + X; X = X + 3. ප්‍රතිදානය “S X”...

A. 12 7
B. 22 13
C. 12 10
D. 10 12

13. ගැලීම් සටහනක් පරිවර්තනයේදී Stop terminal එක වන්නේ...

A. END
B. DISPLAY
C. ENDIF
D. INPUT

14. තීරණයක (ලූපයක් නැති) ඔව් ශාඛාව යන්නේ...

A. END ට පසුව
B. ENDIF ට පසුව
C. ELSE ට පසුව
D. THEN ට පසුව

15. හොඳ ගැලීම් සටහනක නීතියක් කුමක්ද?

A. සෑම දියමන්තියකින්ම පිටවීම් තුනක්
B. එක් Start සහ එක් Stop
C. කොටු තුළ දිගු ඡේද
D. ලේබල් නැතිව ඕනෑම දිශාවකට ඊතල

ව්‍යුහගත ප්‍රශ්න

1. (i) terminal (ii) ආදාන/ප්‍රතිදාන (iii) ක්‍රියාවලිය (iv) තීරණය සඳහා ගැලීම් සටහන් සංකේත ඇඳ, එක් එකක් භාවිත කරන්නේ කුමකටදැයි සඳහන් කරන්න. (ලකුණු 4) *(ලකුණු 4)*

> වටකුරු සෘජුකෝණාස්‍රය – Start/Stop; සමාන්තරාස්‍රය – INPUT/DISPLAY; සෘජුකෝණාස්‍රය – ගණනය/පැවරීම; දියමන්තිය – ඔව්/නැත ප්‍රශ්නය (එකට 1).

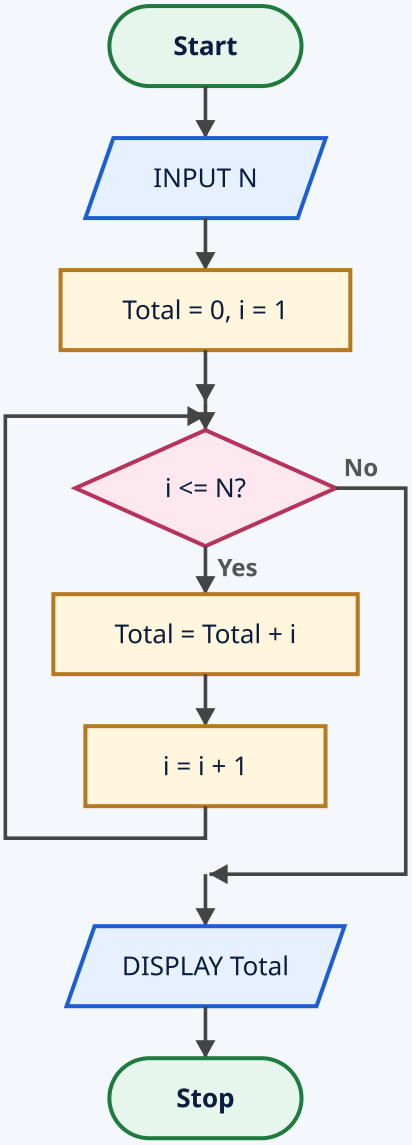
2. සංඛ්‍යා දෙකක් ආදානය කර විශාල එක පෙන්වන ව්‍යාජ කේතය ලියන්න. (ලකුණු 4) *(ලකුණු 4)*

> BEGIN; INPUT A, B; IF A > B THEN Largest = A ELSE Largest = B ENDIF; DISPLAY Largest; END (ආදානය 1, IF 2, පෙන්වීම 1).
> **ව්‍යාජ කේතය (Pseudocode)**

```
BEGIN
  INPUT A, B
  IF A > B THEN
    DISPLAY A
  ELSE
    DISPLAY B
  ENDIF
END
```

3. 1 සිට N දක්වා සංඛ්‍යාවල එකතුව සොයන ගැලීම් සටහනක් අඳින්න. (ලකුණු 5) *(ලකුණු 5)*

Start; N ආදානය; Total = 0, i = 1; දියමන්තිය i <= N?; ඔව් → Total = Total + i, i = i + 1, ආපසු ඊතලය; නැත → Total පෙන්වීම; Stop (ආරම්භ/අවසාන, ආදානය, ආරම්භක අගය, ආපසු ඊතලය සහිත ලූපය, පෙන්වීම එකට 1).



ගැලීම් සටහන – 1 සිට N දක්වා සංඛ්‍යාවල එකතුව

4. S = 0, X = 1; WHILE X < 10: S = S + X; X = X + 3; DISPLAY S, X සඳහා අනුරේඛන වගුවක් සම්පූර්ණ කරන්න. (ලකුණු 4) *(ලකුණු 4)*

S: 0, 1, 5, 12; X: 1, 4, 7, 10; කොන්දේසිය ඔව්, ඔව්, ඔව්, නැත; ප්‍රතිදානය 12 10 (එකට 1).

පියවර	S	X	X < 10 ?
start	0	1	–
loop	1	4	ඔව්
loop	5	7	ඔව්
loop	12	10	ඔව්
check	12	10	නැත

5. ගැලීම් සටහනකට වඩා ව්‍යාජ කේතයේ වාසි දෙකක් සහ ගැලීම් සටහනක එක් වාසියක් දෙන්න. (ලකුණු 3) *(ලකුණු 3)*

ව්‍යාජ කේතය: ලිවීමට සහ වෙනස් කිරීමට ඉක්මන්; සැබෑ කේතයට සමීප (2). ගැලීම් සටහන: රූපයක් ලෙස ගැලීම පහසුවෙන් පෙනේ (1).

සාරාංශය

- ගැලීම් සටහන = රූපය; ව්‍යාජ කේතය = ඉංග්‍රීසි වැනි පේළි.
- Terminal (වටකුරු) = Start/Stop.
- සමාන්තරාස්‍රය = ආදාන/ප්‍රතිදාන.
- සෘජුකෝණාස්‍රය = ක්‍රියාවලිය; දියමන්තිය = තීරණය.
- දියමන්තියකට ඔව් සහ නැත පිටවීම් ඇත.
- දියමන්තියට ආපසු ඊතලය = ලූපය.
- කොටුවෙන් කොටුව, පේළියෙන් පේළිය පරිවර්තනය කරන්න.
- අනුරේඛන වගුවක් ඇල්ගොරිතමයක් පරීක්ෂා කරයි (dry run).

MCQ පිළිතුරු – 1.3

1. **A** – Terminal = වටකුරු.
2. **C** – ගණනය = ක්‍රියාවලිය.
3. **A** – ප්‍රතිදානය = සමාන්තරාස්‍රය.
4. **B** – ඔව් සහ නැත.
5. **C** – සම්බන්ධකය.
6. **D** – පරිගණකය ධාවනය නොකරයි.
7. **C** – ආපසු ඊතලය = පුනරාවර්තනය.
8. **D** – Dry run.
9. **B** – 1+2+3+4+5 = 15.
10. **A** – i <= 5 අසත්‍ය වූ විට නවතී.
11. **B** – 72 > 45.

12. **C** – S=1,5,12; X=4,7,10.
13. **A** – Stop → END.
14. **D** – IF cond THEN ඔව් ELSE නැත.
15. **B** – එක් Start, එක් Stop.

© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.

G.C.E. O/L ICT · 11 ශ්‍රේණිය · පාඩම 1

1.4 හඳුන්වන, විචල්‍ය, නියත සහ දත්ත වර්ග

විෂය නිර්දේශය 10.4. වැඩසටහනකට දත්ත මතක තබා ගත යුතුය – නමක්, ලකුණක්, මිලක්. එය එක් එක් අගය **මතකයේ නමක් ඇති පෙට්ටියක** තබා ගනී. අගය වෙනස් විය හැකි නම් එම පෙට්ටිය **විචල්‍යයකි** (variable); කිසි විටෙක වෙනස් නොවේ නම් එය **නියතයකි** (constant). පෙට්ටියේ නම **හඳුන්වනයයි** (identifier), එහි ඇති අගයේ වර්ගය එහි **දත්ත වර්ගයයි** (data type). සමහර වචන **වෙන් කර ඇති** (reserved) නිසා නම් ලෙස භාවිත කළ නොහැක.

1. විචල්‍ය (Variables)

විචල්‍යයක් යනු වැඩසටහන ධාවනය වන අතරතුර අගය **වෙනස් විය හැකි**, නමක් ඇති මතක ස්ථානයකි. ලේබලයක් ඇති පෙට්ටියක් ගැන සිතන්න: ලේබලය “Marks”, අද ඇතුළත 65; පසුව එම පෙට්ටියටම 80 දැමිය හැක. පැරණි අගය ප්‍රතිස්ථාපනය වේ.

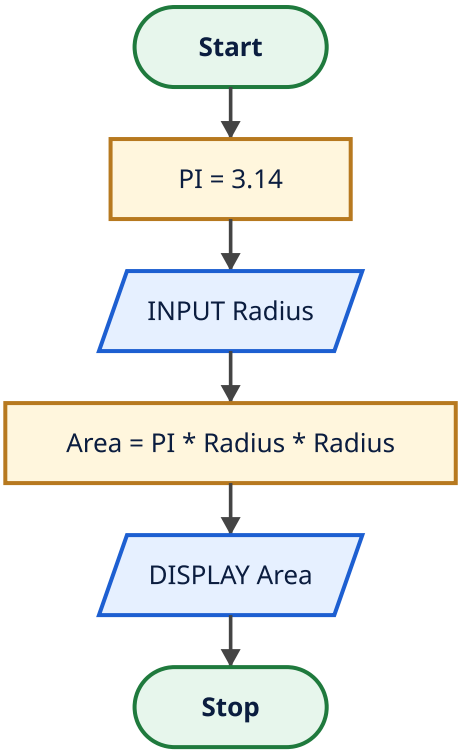
පියවර	ව්‍යාජ කේතය	Marks හි අගය
1	Marks = 65	65
2	Marks = 80	80
3	Marks = Marks + 5	85

2. නියත (Constants)

නියතයක් යනු π = 3.14 හෝ සමත් ලකුණ 35 වැනි, වැඩසටහන ධාවනයේදී **වෙනස් නොවන** නමක් ඇති අගයකි. නියතයක් භාවිතයෙන් වැඩසටහන **කියවීමට පහසු** සහ **වෙනස් කිරීමට පහසු** වේ – සමත් ලකුණ වෙනස් වුවහොත් එක් තැනක පමණක් වෙනස් කළ යුතුය. Pascal හි එය **const** යටතේ ලියමු. Python හි සැබෑ නියත නැති නිසා සම්ප්‍රදායිකව නම ලොකු අකුරින් ලියමු.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  CONSTANT PI = 3.14
  INPUT Radius
  Area = PI * Radius * Radius
  DISPLAY Area
END
```



ගැලීම් සටහන - නියතයක්: වෘත්තයක වර්ගඵලය

නියතයක්: වෘත්තයක වර්ගඵලය

Pascal

```
program CircleArea;
const
  PI = 3.14;
var
  radius, area: real;
begin
  write('Enter radius: ');
  readln(radius);
  area := PI * radius * radius;
  writeln('Area = ', area:0:2);
end.
```

Python

```
PI = 3.14            # a constant (capital letters by custom)
radius = float(input("Enter radius: "))
area = PI * radius * radius
print(f"Area = {area:.2f}")
```

නියැදි ධාවනය

```
Enter radius: 7
Area = 153.86
```

3. දත්ත වර්ග

දත්ත වර්ගය මගින් විචල්‍යයක ඇති අගයේ වර්ගය සහ එයට අවශ්‍ය මතක ප්‍රමාණය පරිගණකයට කියයි. O/L විෂය නිර්දේශය මූලික වර්ග හතරක් භාවිත කරයි:

දත්ත වර්ගය	දරන්නේ	උදාහරණ	Pascal	Python
පූර්ණ සංඛ්‍යා	පූර්ණ සංඛ්‍යා	16, −5, 2026	integer	int
දශම සංඛ්‍යා (real)	දශම තිතක් ඇති සංඛ්‍යා	1.62, 3.14, 71.67	real	float
අක්ෂර	එක් අකුරක්, ඉලක්කමක් හෝ සංකේතයක්	'A', '7', '#'	char	දිග 1 වන str
තාර්කික (Boolean)	සත්‍ය හෝ අසත්‍ය පමණි	True, False	boolean	bool
String (අමතර)	පාඨය – අක්ෂර බොහොමයක්	'Nimal', 'Kandy'	string	str

විවිධ දත්ත වර්ගවල විචල්‍ය

Pascal

```
program DataTypes;
var
  name: string;
  age: integer;
  height: real;
  grade: char;
  isPrefect: boolean;
begin
  name := 'Nimal';
  age := 16;
  height := 1.62;
  grade := 'A';
  isPrefect := true;
  writeln('Name: ', name);
  writeln('Age: ', age);
  writeln('Height: ', height:0:2);
  writeln('Grade: ', grade);
  if isPrefect then
    writeln('Prefect: Yes')
  else
    writeln('Prefect: No');
end.
```

Python

```
name = "Nimal"        # string
age = 16              # integer
height = 1.62         # floating point (real)
grade = "A"           # character
is_prefect = True     # logical (Boolean)
print("Name:", name)
print("Age:", age)
print(f"Height: {height:.2f}")
print("Grade:", grade)
if is_prefect:
    print("Prefect: Yes")
else:
    print("Prefect: No")
```

නියැදි ධාවනය

```
Name: Nimal
Age: 16
Height: 1.62
Grade: A
Prefect: Yes
```

💡 **Pascal** හි සෑම විචල්‍යයක්ම භාවිතයට පෙර එහි වර්ගය සමඟ **ප්‍රකාශ කළ** (declare) යුතුය (var age: integer;). **Python** හි අගයක් පැවරීම පමණක් ප්‍රමාණවත් (age = 16), වර්ගය Python විසින්ම සොයා ගනී. දෙකම හරි – දත්ත වර්ගයේ අදහස එකමය.

4. හඳුන්වන සහ නම් කිරීමේ නීති

හඳුන්වනයක් යනු විචල්‍යයකට, නියතයකට, වැඩසටහනකට හෝ උප වැඩසටහනකට දෙන නමයි. නීති (Pascal සහ Python):

- අකුරු, ඉලක්කම් සහ යටි ඉර _ පමණක් භාවිත කරන්න.
- එය **අකුරකින්** (හෝ _) **ආරම්භ විය යුතුය**, කිසි විටෙක ඉලක්කමකින් නොවේ.
- **හිස්තැන් නැත**, -, %, @ වැනි සංකේත නැත.
- එය **වෙන් කළ වචනයක් නොවිය යුතුය**.
- Python **අකුරු වර්ගයට සංවේදීය** (Total සහ total වෙනස්). Pascal **සංවේදී නොවේ** (Total සහ TOTAL එකම නමකි).

හඳුන්වනය	වලංගුද?	හේතුව
total_marks	ඔව්	අකුරු සහ _
Mark2	ඔව්	ඉලක්කම මුලින් නොවේ
2ndMark	නැත	ඉලක්කමකින් ආරම්භ වේ
total marks	නැත	හිස්තැනක් ඇත
pass%	නැත	සංකේතයක් ඇත
begin / if	නැත	වෙන් කළ වචනයකි

5. වෙන් කළ වචන (keywords)

වෙන් කළ වචන යනු භාෂාවේ දැනටමත් විශේෂ අර්ථයක් ඇති වචන වන නිසා ඒවා හඳුන්වන ලෙස භාවිත කළ නොහැක. Pascal: program, var, const, begin, end, if, then, else, for, while, repeat, until, and, or, not, div, mod. Python: if, else, elif, for, while, def, return, True, False, and, or, not, import.

6. අර්ථවත් නම්

අගය කුමක්දැයි **කියන** නම් තෝරන්න. x = a * b සහ total = price * quantity සසඳන්න. දෙවැන්න තේරුම් ගැනීමට සහ පසුව නිවැරදි කිරීමට බෙහෙවින් පහසුය. හොඳ: studentName, totalMarks, unit_price. දුර්වල: a, x1, abc.

> 💡 Pascal උගුල: Pascal ලොකු/කුඩා අකුරු නොසලකන නිසා **Discount** නම් වැඩසටහනකට **discount** නම් විචල්‍යයක්ද තිබිය නොහැක – සම්පාදකය “Duplicate identifier” කියයි. එබැවින් 1.6 හි අපේ වට්ටම් වැඩසටහන **BillDiscount** ලෙස නම් කර ඇත.

ප්‍රධාන පද

විචල්‍යය – අගය වෙනස් විය හැකි නමක් ඇති මතක ස්ථානයක්.

නියතය – වැඩසටහන ධාවනයේදී වෙනස් නොවන නමක් ඇති අගයක්.

හඳුන්වනය – විචල්‍යයක, නියතයක, වැඩසටහනක හෝ උප වැඩසටහනක නම.

වෙන් කළ වචනය – භාෂාවේ විශේෂ අර්ථයක් ඇති වචනයක්; නමක් ලෙස ඉඩ නැත.

දත්ත වර්ගය – විචල්‍යයක ඇති අගයේ වර්ගය.

පූර්ණ සංඛ්‍යාව – 16 හෝ −5 වැනි පූර්ණ සංඛ්‍යාවක්.

දශම සංඛ්‍යාව – 1.62 වැනි දශම තිතක් ඇති සංඛ්‍යාවක් (Pascal: real).

අක්ෂරය – 'A' වැනි තනි අකුරක්, ඉලක්කමක් හෝ සංකේතයක්.

තාර්කික (Boolean) – සත්‍ය සහ අසත්‍ය යන අගයන් දෙක පමණක් ඇති වර්ගයක්.

ප්‍රකාශනය – භාවිතයට පෙර විචල්‍යයක නම සහ වර්ගය පරිගණකයට කීම.

අකුරු සංවේදී – ලොකු සහ කුඩා අකුරු වෙනස් ලෙස සලකයි (Python).

අර්ථවත් නම – අගය කුමක්දැයි පෙන්වන නමක්, උදා totalMarks.

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. විචල්‍යයක් සහ නියතයක් අතර වෙනස කුමක්ද?

පිළිතුර: විචල්‍යයක අගය ධාවනයේදී වෙනස් විය හැක; නියතයක අගය වෙනස් විය නොහැක.

2. විෂය නිර්දේශයේ මූලික දත්ත වර්ග හතර නම් කරන්න.

පිළිතුර: පූර්ණ සංඛ්‍යා, දශම (real), අක්ෂර සහ තාර්කික (Boolean).

3. 2ndMark වලංගු හඳුන්වනයක්ද? ඇයි?

පිළිතුර: නැත – හඳුන්වනයක් ඉලක්කමකින් ආරම්භ විය නොහැක.

4. 16, 1.62, 'A', True සඳහා දත්ත වර්ගය දෙන්න.

පිළිතුර: පූර්ණ සංඛ්‍යා, දශම, අක්ෂර, තාර්කික.

5. අර්ථවත් නම් භාවිත කළ යුත්තේ ඇයි?

පිළිතුර: වැඩසටහන කියවීමට, තේරුම් ගැනීමට සහ නිවැරදි කිරීමට පහසුය.

6. වෙන් කළ වචනයක් යනු කුමක්ද? උදාහරණ දෙකක් දෙන්න.

පිළිතුර: භාෂාවේ විශේෂ අර්ථයක් ඇති වචනයක්, උදා if, while (begin, end, for...).

බහුවරණ ප්‍රශ්න

1. අගය වෙනස් විය හැකි නමක් ඇති මතක ස්ථානයක්...

A. විචල්‍යයකි
B. වෙන් කළ වචනයකි
C. විවරණයකි
D. නියතයකි

2. කිසි විටෙක වෙනස් නොවන π සඳහා 3.14 ගබඩා කිරීමට වඩාත් සුදුසු...

A. අක්ෂරයකි
B. නියතයකි
C. විචල්‍යයකි
D. Boolean එකකි

3. වලංගු හඳුන්වනයක් කුමක්ද?

A. 2total
B. total marks
C. total_marks
D. total-marks

4. වලංගු හඳුන්වනයක් නොවන්නේ කුමක්ද?

A. Mark2
B. unit_price
C. begin
D. studentName

5. 1.62 හි දත්ත වර්ගය...

A. පූර්ණ සංඛ්‍යා
B. අක්ෂර
C. තාර්කික
D. දශම (real)

6. 'A' හි දත්ත වර්ගය...

A. පූර්ණ සංඛ්‍යා
B. real
C. අක්ෂර
D. Boolean

7. තාර්කික (Boolean) විචල්‍යයකට දැරිය හැක්කේ...

A. දශම පමණි
B. ඕනෑම පාඨයක්
C. ඕනෑම පූර්ණ සංඛ්‍යාවක්
D. සත්‍ය හෝ අසත්‍ය පමණි

8. පන්තියක සිසුන් ගණන ගබඩා කිරීමට වඩාත් සුදුසු...

A. පූර්ණ සංඛ්‍යා
B. real
C. Boolean
D. අක්ෂර

9. Pascal හි දශම සංඛ්‍යාවක් සඳහා වර්ගය...

A. char
B. real
C. int
D. float

10. Python හි පූර්ණ සංඛ්‍යාවක් සඳහා වර්ගය...

A. int
B. real
C. integer
D. bool

11. අකුරු වර්ගයට සංවේදී (Total ≠ total) භාෂාව කුමක්ද?

A. Pascal
B. දෙකම
C. කිසිවක් නැත
D. Python

12. එක් භාණ්ඩයක මිල සඳහා වඩාත්ම අර්ථවත් නම කුමක්ද?

A. abc
B. a1
C. unitPrice
D. x

13. Marks = 65, Marks = 80, Marks = Marks + 5 ට පසු Marks හි අගය...

A. 85
B. 150
C. 65
D. 70

14. Pascal හි විචල්‍ය ප්‍රකාශ කරන්නේ ... වචනය යටතේය

A. const
B. var
C. def
D. begin

15. අරය 7 සහ PI = 3.14 සඳහා වෘත්ත වැඩසටහන පෙන්වන්නේ...

A. Area = 21.98
B. Area = 153.86
C. Area = 43.96
D. Area = 49.00

ව්‍යුහගත ප්‍රශ්න

1. (i) විචල්‍යයක් සහ නියතයක් අර්ථ දක්වන්න. (ලකුණු 2)
(ii) පාසල් ගාස්තු වැඩසටහනකින් එක් එකකට උදාහරණයක් දෙන්න. (ලකුණු 2) *(ලකුණු 4)*

> (i) විචල්‍යය – අගය වෙනස් විය හැකි නමක් ඇති මතක ස්ථානය; නියතය – වෙනස් නොවන නමක් ඇති අගය. (ii) විචල්‍යය: සිසුන් ගණන / ගෙවූ මුදල; නියතය: වාරයකට ගාස්තුව (උදා 500).

2. පහත සඳහා වඩාත් සුදුසු දත්ත වර්ගය දෙන්න: (a) සිසුවෙකුගේ වයස (b) මීටර්වලින් උස (c) ශ්‍රේණි අකුර (d) සිසුවා ශිෂ්‍ය නායකයෙක්ද යන්න. (ලකුණු 4) *(ලකුණු 4)*

> (a) පූර්ණ සංඛ්‍යා (b) දශම / real (c) අක්ෂර (d) තාර්කික / Boolean.

3. එක් එක් හඳුන්වනය වලංගුද යන්න හේතුව සමඟ සඳහන් කරන්න: total, 3sum, my mark, if, Avg_1. (ලකුණු 5) *(ලකුණු 5)*

> total – වලංගු; 3sum – අවලංගු, ඉලක්කමකින් ආරම්භය; my mark – අවලංගු, හිස්තැන; if – අවලංගු, වෙන් කළ වචනය; Avg_1 – වලංගු.

4. සෑම තැනම 35 ටයිප් කිරීම වෙනුවට PASSMARK = 35 වැනි නියතයක් භාවිතයේ වාසි දෙකක් පහදන්න. (ලකුණු 2) *(ලකුණු 2)*

> කියවීමට පහසුයි (නම මගින් අර්ථය කියයි) (1); වෙනස් කිරීමට පහසුයි – එක් තැනක පමණක් වෙනස් කරයි (1).

5. Discount නම් Pascal වැඩසටහනක discount නම් විචල්‍යයක් ඇත. එය compile නොවන්නේ ඇයි? නිවැරදි කරන්නේ කෙසේද? (ලකුණු 2) *(ලකුණු 2)*

> Pascal අකුරු වර්ගයට සංවේදී නොවන නිසා නම් දෙකම එකම හඳුන්වනයයි (“Duplicate identifier”) (1). වැඩසටහන නැවත නම් කරන්න, උදා BillDiscount (1).

සාරාංශය

- විචල්‍යය: අගය වෙනස් විය හැක.
- නියතය: අගය ස්ථාවරයි.
- හඳුන්වනය = නමක්; අකුරකින් ආරම්භය, හිස්තැන් නැත.
- වෙන් කළ වචන නම් ලෙස භාවිත කළ නොහැක.
- පූර්ණ, දශම, අක්ෂර, තාර්කික.

- Pascal: integer, real, char, boolean.
- Python: int, float, str, bool; අකුරු සංවේදී.
- totalMarks වැනි අර්ථවත් නම් භාවිත කරන්න.

MCQ පිළිතුරු – 1.4

1. **A** – විචල්‍යය.
2. **B** – ස්ථාවර අගය = නියතය.
3. **C** – අකුරු සහ යටි ඉර පමණි.
4. **C** – begin Pascal හි වෙන් කළ වචනයකි.
5. **D** – දශම තිතක් ඇත.
6. **C** – තනි අකුර.
7. **D** – අගයන් දෙකක් පමණි.
8. **A** – පූර්ණ සංඛ්‍යාව.
9. **B** – Pascal real භාවිත කරයි.
10. **A** – Python int භාවිත කරයි.
11. **D** – Python සංවේදීයි; Pascal නොවේ.
12. **C** – එය කුමක්දැයි කියයි.
13. **A** – 80 + 5.
14. **B** – var කොටස.
15. **B** – 3.14 × 7 × 7 = 153.86.

© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.

G.C.E. O/L ICT · 11 ශ්‍රේණිය · පාඩම 1

1.5 කාරක: ගණිත, සංසන්දන සහ තාර්කික

විෂය නිර්දේශය 10.5. කාරක (operators) යනු අගයන් මත වැඩ කරන සංකේතයි. **ගණිත** කාරක ගණනය කරයි (+ − × ÷). **සංසන්දන** කාරක අගයන් දෙකක් සසඳා සත්‍ය හෝ අසත්‍ය දෙයි. **තාර්කික** කාරක කොන්දේසි සම්බන්ධ කරයි (AND, OR, NOT). එක් පේළියක කාරක බොහොමයක් ඇති විට, මුලින් කරන්නේ කුමක්දැයි **ප්‍රමුඛතාව** (precedence) තීරණය කරයි – ගණිතයේ BODMAS මෙන්ම.

1. ගණිත කාරක

මෙහෙයුම	Pascal	Python	17 සහ 5
එකතු කිරීම	+	+	22
අඩු කිරීම	-	-	12
ගුණ කිරීම	*	*	85
බෙදීම (දශම පිළිතුර)	/	/	3.40
පූර්ණ සංඛ්‍යා බෙදීම (ලබ්ධිය)	div	//	3
ශේෂය (modulus)	mod	%	2

අංක ගණිත කාරක

Pascal

```
program Operators;
var
  a, b: integer;
begin
  a := 17;
  b := 5;
  writeln('Sum = ', a + b);
  writeln('Difference = ', a - b);
  writeln('Product = ', a * b);
  writeln('Division = ', a / b:0:2);
  writeln('Quotient = ', a div b);
  writeln('Remainder = ', a mod b);
end.
```

Python

```
a = 17
b = 5
print("Sum =", a + b)
print("Difference =", a - b)
print("Product =", a * b)
print(f"Division = {a / b:.2f}")
print("Quotient =", a // b)
print("Remainder =", a % b)
```

නියැදි ධාවනය

```
Sum = 22
Difference = 12
Product = 85
Division = 3.40
Quotient = 3
Remainder = 2
```

💡 **mod** ඉතා ප්‍රයෝජනවත්ය: N mod 2 = 0 නම් N **ඉරට්ටේ** වේ. 17 = 3 × 5 + 2 නිසා 17 mod 5 = 2.

2. සංසන්දන කාරක

සංසන්දනයක් සැමවිටම **තාර්කික** ප්‍රතිඵලයක් දෙයි: සත්‍ය හෝ අසත්‍ය. ප්‍රවේශම් වන්න – Pascal සහ Python “සමාන” සහ “අසමාන” වෙනස් ලෙස ලියයි.

තේරුම	Pascal / ව්‍යාජ කේතය	Python	උදාහරණය (A = 8, B = 5)
සමාන	=	==	A = B → අසත්‍ය
අසමාන	<>	!=	A <> B → සත්‍ය
වැඩි	>	>	A > B → සත්‍ය
අඩු	<	<	A < B → අසත්‍ය

වැඩි හෝ සමාන	>=	>=	A >= 8 → සත්‍ය
අඩු හෝ සමාන	<=	<=	B <= 4 → අසත්‍ය

3. තාර්කික කාරක

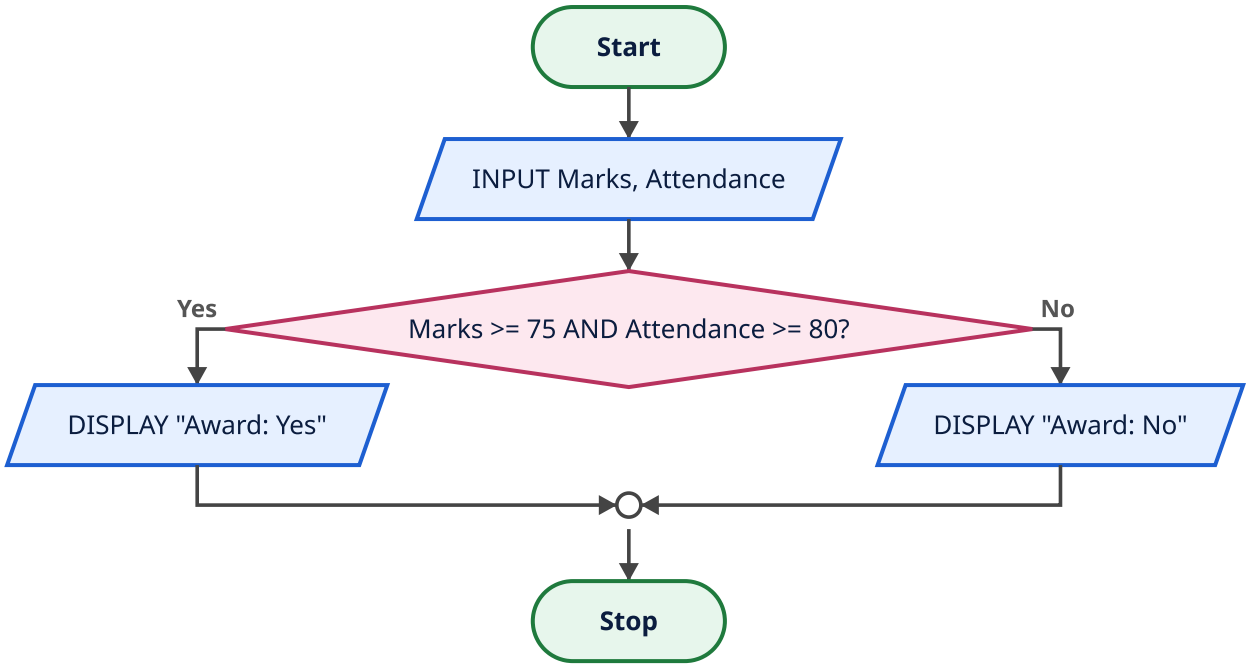
AND සත්‍ය වන්නේ කොන්දේසි **දෙකම** සත්‍ය වූ විට පමණි. **OR** සත්‍ය වන්නේ **අවම වශයෙන් එකක්** සත්‍ය වූ විටය. **NOT** අගය ප්‍රතිවිරුද්ධ කරයි. (10 ශ්‍රේණියේ තාර්කික ද්වාර ලෙස මේවා හමු විය.)

A	B	A AND B	A OR B	NOT A
සත්‍ය	සත්‍ය	සත්‍ය	සත්‍ය	අසත්‍ය
සත්‍ය	අසත්‍ය	අසත්‍ය	සත්‍ය	අසත්‍ය
අසත්‍ය	සත්‍ය	අසත්‍ය	සත්‍ය	සත්‍ය
අසත්‍ය	අසත්‍ය	අසත්‍ය	අසත්‍ය	සත්‍ය

උදාහරණය: ශිෂ්‍යත්වයක් දෙන්නේ ලකුණු ≥ 75 **AND** පැමිණීම ≥ 80% නම් පමණි.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT Marks, Attendance
  IF Marks >= 75 AND Attendance >= 80 THEN
    DISPLAY "Award: Yes"
  ELSE
    DISPLAY "Award: No"
  ENDIF
END
```



ගැලීම් සටහන – තාර්කික කාරකය AND: ශිෂ්‍යත්ව ත්‍යාගය

තාර්කික කාරකය AND: ශිෂ්‍යත්ව ත්‍යාගය

Pascal

```
program Award;
var
  marks, attendance: integer;
begin
  write('Enter marks: ');
  readln(marks);
  write('Enter attendance %: ');
  readln(attendance);
  if (marks >= 75) and (attendance >= 80) then
    writeln('Award: Yes')
  else
    writeln('Award: No');
end.
```

Python

```
marks = int(input("Enter marks: "))
attendance = int(input("Enter attendance %: "))
if marks >= 75 and attendance >= 80:
    print("Award: Yes")
else:
    print("Award: No")
```

නියැදි ධාවනය 1

```
Enter marks: 80
Enter attendance %: 85
Award: Yes
```

නියැදි ධාවනය 2

```
Enter marks: 80
Enter attendance %: 70
Award: No
```

4. කාරක ප්‍රමුඛතාව

පිළිවෙළ	කාරක
1	( ) වරහන්
2	NOT
3	* / div mod (Python: * / // %)
4	+ −
5	සංසන්දන = <> < > <= >=
6	AND, පසුව OR

එකම මට්ටමේ කාරක **වමේ සිට දකුණට** කරනු ලැබේ. එබැවින් 20 − 8 / 4 × 2 = 20 − 2 × 2 = 20 − 4 = 16, සහ 10 − 4 − 3 = 6 − 3 = 3. සැකයක් ඇත්නම් **වරහන් භාවිත කරන්න** – ඒවා පිළිවෙළ සැමට පැහැදිලි කරයි.

කාරක ප්‍රමුඛතාව

Pascal

```
program Precedence;
begin
  writeln('2 + 3 * 4 = ', 2 + 3 * 4);
  writeln('(2 + 3) * 4 = ', (2 + 3) * 4);
  writeln('20 - 8 / 4 * 2 = ', 20 - 8 / 4 * 2:0:2);
  writeln('10 - 4 - 3 = ', 10 - 4 - 3);
end.
```

Python

```
print("2 + 3 * 4 =", 2 + 3 * 4)
print("(2 + 3) * 4 =", (2 + 3) * 4)
print("20 - 8 / 4 * 2 =", f"{20 - 8 / 4 * 2:.2f}")
print("10 - 4 - 3 =", 10 - 4 - 3)
```

නියැදි ධාවනය

```
2 + 3 * 4 = 14
(2 + 3) * 4 = 20
20 - 8 / 4 * 2 = 16.00
10 - 4 - 3 = 3
```

💡 Pascal සටහන: Pascal හි AND සහ OR සංසන්දනයට **පෙර** කරන නිසා වරහන් දැමිය යුතුය: if (Marks >= 75) and (Att >= 80) then … Python ට ඒවා අවශ්‍ය නැත, නමුත් වරහන් කිසිදා හානියක් නොකරයි.

ප්‍රධාන පද

කාරකය – අගයන් මත වැඩ කරන සංකේතයක්, උදා +.

කාර්යාංකය – කාරකයක් වැඩ කරන අගයක්, උදා 17 + 5 හි 17 සහ 5.

ගණිත කාරකය – ගණනය කරයි: + − * / div mod.

div / // – පූර්ණ සංඛ්‍යා බෙදීම: 17 div 5 = 3.

mod / % – බෙදීමෙන් පසු ශේෂය: 17 mod 5 = 2.

සංසන්දන කාරකය – අගයන් දෙකක් සසඳා සත්‍ය හෝ අසත්‍ය දෙයි.

අසමාන – Pascal හි <>, Python හි !=.

තාර්කික කාරකය – කොන්දේසි සම්බන්ධ කරයි හෝ ප්‍රතිවිරුද්ධ කරයි: AND, OR, NOT.

AND – කොන්දේසි දෙකම සත්‍ය නම් පමණක් සත්‍ය.

OR – අවම වශයෙන් එකක් සත්‍ය නම් සත්‍ය.

NOT – සත්‍ය අසත්‍ය ද, අසත්‍ය සත්‍ය ද කරයි.

ප්‍රමුඛතාව – කාරක ක්‍රියාත්මක වන පිළිවෙළ.

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. 17 div 5 සහ 17 mod 5 සොයන්න.

> **පිළිතුර:** 3 සහ 2.

2. Pascal සහ Python හි “අසමාන” ලියන්නේ කෙසේද?

> **පිළිතුර:** Pascal <>, Python !=.

3. A AND B සත්‍ය වන්නේ කවදාද?

> **පිළිතුර:** A සහ B දෙකම සත්‍ය වූ විට පමණි.

4. 2 + 3 * 4 ගණනය කරන්න.

> **පිළිතුර:** 14 (මුලින් ගුණ කිරීම).

5. 20 − 8 / 4 * 2 ගණනය කරන්න.

> **පිළිතුර:** 16 (8 / 4 = 2, 2 × 2 = 4, 20 − 4 = 16).

6. N ඉරට්ටේදැයි පරීක්ෂා කරන්නේ කෙසේද?

> **පිළිතුර:** N mod 2 = 0 (Python: N % 2 == 0).

බහුවරණ ප්‍රශ්න

1. 17 mod 5...

A. 12
B. 3
C. 3.4
D. 2

2. 17 div 5...

A. 85
B. 3.4
C. 3
D. 2

3. Python හි පූර්ණ සංඛ්‍යා බෙදීම ලියන්නේ...

A. //
B. /
C. div
D. %

4. Python හි ශේෂ කාරකය...

A. #
B. %
C. //
D. mod

5. Pascal හි “අසමාන” ලියන්නේ...

A. <>
B. ==
C. !=
D. =<

6. Python හි සංසන්දනයක “සමාන” ලියන්නේ...

A. <>
B. :=
C. ==
D. =

7. සංසන්දන කාරකයක් සැමවිටම දෙන්නේ...

A. දශමයක්
B. පාඨයක්
C. සත්‍ය හෝ අසත්‍ය
D. අක්ෂරයක්

8. True AND False දෙන්නේ...

A. දෝෂය
B. 1
C. සත්‍ය
D. අසත්‍ය

9. True OR False දෙන්නේ...

A. අසත්‍ය
B. 0
C. සත්‍ය
D. දෝෂය

10. NOT True දෙන්නේ...

A. අසත්‍ය
B. 0.5
C. කිසිවක් නැත
D. සත්‍ය

11. 2 + 3 * 4 =

A. 9
B. 14
C. 24
D. 20

12. (2 + 3) * 4 =

A. 9
B. 20
C. 24
D. 14

13. 10 − 4 − 3 =

A. 3
B. 9
C. 17
D. −3

14. ලකුණු = 80, පැමිණීම = 70. “Marks >= 75 AND attendance >= 80”...

A. සත්‍ය
B. 80
C. දෝෂය
D. අසත්‍ය

15. මුලින්ම කරන්නේ කුමන කාරකයද?

A. +
B. ( ) වරහන්
C. AND
D. >=

ව්‍යුහගත ප්‍රශ්න

1. A = 17 සහ B = 5 භාවිතයෙන් සොයන්න: (a) A + B (b) A / B (c) A div B (d) A mod B. (ලකුණු 4) *(ලකුණු 4)*

(a) 22 (b) 3.4 (c) 3 (d) 2.

අංක ගණිත කාරක

Pascal

```
program Operators;
var
  a, b: integer;
begin
  a := 17;
  b := 5;
  writeln('Sum = ', a + b);
  writeln('Difference = ', a - b);
  writeln('Product = ', a * b);
  writeln('Division = ', a / b:0:2);
  writeln('Quotient = ', a div b);
  writeln('Remainder = ', a mod b);
end.
```

Python

```
a = 17
b = 5
print("Sum =", a + b)
print("Difference =", a - b)
print("Product =", a * b)
print(f"Division = {a / b:.2f}")
print("Quotient =", a // b)
print("Remainder =", a % b)
```

නියැදි ධාවනය

```
Sum = 22
Difference = 12
Product = 85
Division = 3.40
Quotient = 3
Remainder = 2
```

2. A AND B, A OR B සහ NOT A සඳහා සත්‍යතා වගුව සම්පූර්ණ කරන්න. (ලකුණු 4) *(ලකුණු 4)*

AND: T,T සඳහා පමණක් T; OR: F,F සඳහා පමණක් F; NOT A: A හි ප්‍රතිවිරුද්ධය (සටහන්වල වගුව බලන්න).

3. පියවර පෙන්වමින් ගණනය කරන්න: (a) 2 + 3 * 4 (b) (2 + 3) * 4 (c) 20 − 8 / 4 * 2. (ලකුණු 3) *(ලකුණු 3)*

(a) 2 + 12 = 14 (b) 5 × 4 = 20 (c) 20 − 2 × 2 = 20 − 4 = 16.

කාරක ප්‍රමුඛතාව

Pascal

```
program Precedence;
begin
  writeln('2 + 3 * 4 = ', 2 + 3 * 4);
  writeln('(2 + 3) * 4 = ', (2 + 3) * 4);
  writeln('20 - 8 / 4 * 2 = ', 20 - 8 / 4 * 2:0:2);
  writeln('10 - 4 - 3 = ', 10 - 4 - 3);
end.
```

Python

```
print("2 + 3 * 4 =", 2 + 3 * 4)
print("(2 + 3) * 4 =", (2 + 3) * 4)
print("20 - 8 / 4 * 2 =", f"{20 - 8 / 4 * 2:.2f}")
print("10 - 4 - 3 =", 10 - 4 - 3)
```

නියැදි ධාවනය

```
2 + 3 * 4 = 14
(2 + 3) * 4 = 20
20 - 8 / 4 * 2 = 16.00
10 - 4 - 3 = 3
```

4. ලකුණු ≥ 75 සහ පැමිණීම ≥ 80 නම් සිසුවාට ශිෂ්‍යත්වයක් ලැබේ. කොන්දේසිය ලියා, ලකුණු 80 / පැමිණීම 85 සහ ලකුණු 80 / පැමිණීම 70 සඳහා ප්‍රතිඵලය දෙන්න. (ලකුණු 3) *(ලකුණු 3)*

(Marks >= 75) AND (Attendance >= 80) (1); 80/85 → ඔව් (1); 80/70 → නැත (1).

5. Python හි = සහ == අතර වෙනස පහදන්න. (ලකුණු 2) *(ලකුණු 2)*

= විචල්‍යයකට අගයක් පවරයි (ගබඩා කරයි) (1); == අගයන් දෙකක් සසඳා සත්‍ය හෝ අසත්‍ය දෙයි (1).

සාරාංශය

- ගණිත: + − * / div mod.
- Python div සඳහා // ද mod සඳහා % ද භාවිත කරයි.
- සංසන්දනය සත්‍ය හෝ අසත්‍ය දෙයි.
- අසමාන: Pascal <>, Python !=.
- AND: දෙකම සත්‍ය; OR: එකක් සත්‍ය; NOT: ප්‍රතිවිරුද්ධය.
- මුලින් වරහන්, පසුව * /, පසුව + −.
- එකම මට්ටම: වමේ සිට දකුණට.

- N mod 2 = 0 නම් N ඉරට්ටේ.

MCQ පිළිතුරු – 1.5

1. **D** – ශේෂය 2.
2. **C** – පූර්ණ ලබ්ධිය.
3. **A** – //.
4. **B** – %.
5. **A** – <>.
6. **C** – == සසඳයි; = පවරයි.
7. **C** – තාර්කික ප්‍රතිඵලය.
8. **D** – AND ට දෙකම සත්‍ය විය යුතුය.
9. **C** – OR ට එකක් සත්‍ය නම් ප්‍රමාණවත්.
10. **A** – NOT ප්‍රතිවිරුද්ධ කරයි.
11. **B** – මුලින් ගුණ කිරීම.
12. **B** – මුලින් වරහන්.
13. **A** – වමේ සිට දකුණට.
14. **D** – දෙවන කොටස අසත්‍යයි.
15. **B** – වරහන් ඉහළම.

© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.

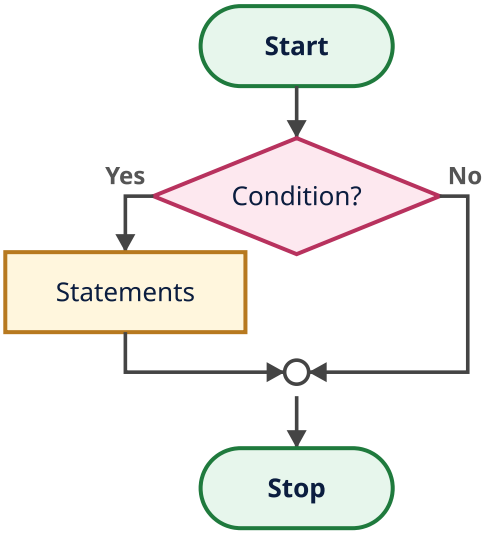
G.C.E. O/L ICT · 11 ශ්‍රේණිය · පාඩම 1

1.6 වරණය: If–EndIf, If–Else–EndIf සහ Case

විෂය නිර්දේශය 10.6. වරණය මගින් වැඩසටහනකට තීරණ ගත හැක. ආකාර තුනක් ඇත: **If–EndIf** (කොන්දේසිය සත්‍ය වූ විට පමණක් යමක් කිරීම), **If–Else–EndIf** (මාර්ග දෙකකින් එකක් තේරීම) සහ **Case** (විකල්ප බොහොමයකින් එකක් තේරීම). එක් එකක් සඳහා **ගැලීම් සටහන → ව්‍යාජ කේතය → Pascal සහ Python කේතය** ලෙස යමු.

1. If–EndIf (ඒක-මාර්ග වරණය)

කොන්දේසිය **සත්‍ය** නම් ඇතුළත ප්‍රකාශ ක්‍රියාත්මක වේ. **අසත්‍ය** නම් ඒවා **මඟ හැරී** ENDIF ට පසුව වැඩසටහන දිගටම යයි.



If–EndIf හි සාමාන්‍ය ආකාරය

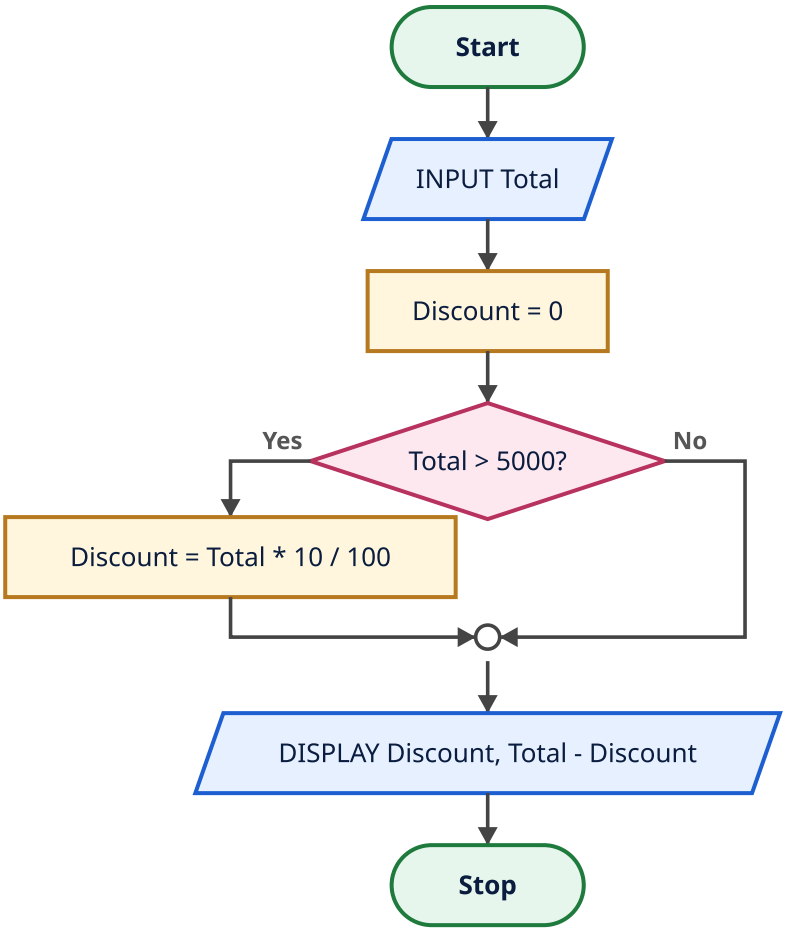
සාමාන්‍ය ව්‍යාජ කේතය

```
IF condition THEN
  statements
ENDIF
```

උදාහරණය: ගාල්ලේ කඩයක් බිල **රු. 5000 ට වැඩි** විට **10% වට්ටමක්** දෙයි.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT Total
  Discount = 0
  IF Total > 5000 THEN
    Discount = Total * 10 / 100
  ENDIF
  DISPLAY Discount, Total - Discount
END
```



ගැලීම් සටහන – If–EndIf: Rs. 5000 ට වැඩි බිලකට 10% වට්ටම

If-EndIf: Rs. 5000 ට වැඩි බිලකට 10% වට්ටම

Pascal

```
program BillDiscount;
var
  total, discount: integer;
begin
  write('Enter bill total (Rs.): ');
  readln(total);
  discount := 0;
  if total > 5000 then
    discount := total * 10 div 100;
  writeln('Discount = Rs. ', discount);
  writeln('Amount to pay = Rs. ', total - discount);
end.
```

Python

```
total = int(input("Enter bill total (Rs.): "))
discount = 0
if total > 5000:
    discount = total * 10 // 100
print("Discount = Rs.", discount)
print("Amount to pay = Rs.", total - discount)
```

නියැදි ධාවනය 1

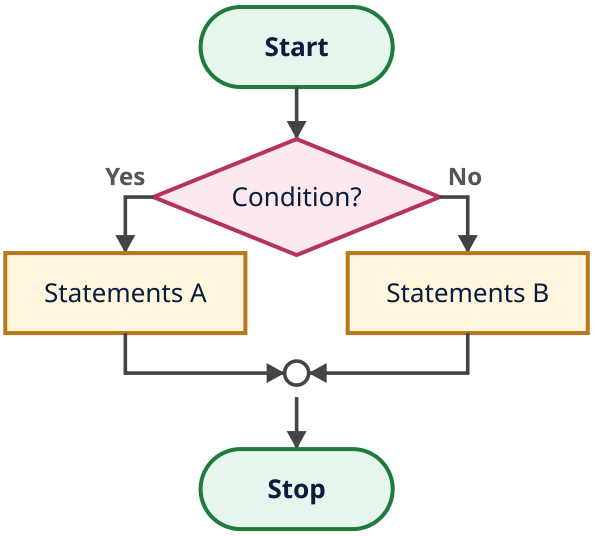
```
Enter bill total (Rs.): 6500
Discount = Rs. 650
Amount to pay = Rs. 5850
```

නියැදි ධාවනය 2

```
Enter bill total (Rs.): 3000
Discount = Rs. 0
Amount to pay = Rs. 3000
```

2. If-Else-EndIf (ද්වි-මාර්ග වරණය)

ප්‍රකාශ කණ්ඩායම් **දෙකෙන් හරියටම එකක්** ක්‍රියාත්මක වේ: කොන්දේසිය සත්‍ය නම් THEN කොටස, අසත්‍ය නම් ELSE කොටස.



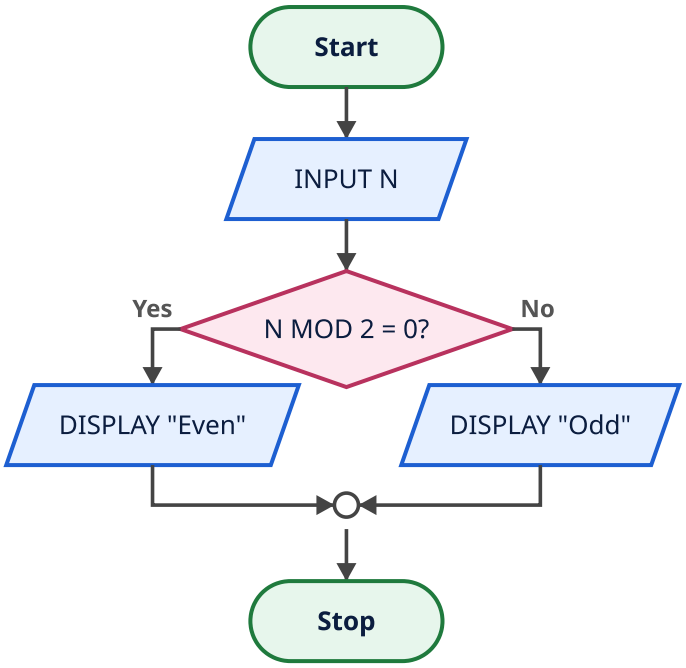
If-Else-EndIf හි සාමාන්‍ය ආකාරය

සාමාන්‍ය ව්‍යාජ කේතය

```
IF condition THEN
  statements A
ELSE
  statements B
ENDIF
```

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT N
  IF N MOD 2 = 0 THEN
    DISPLAY "Even"
  ELSE
    DISPLAY "Odd"
  ENDIF
END
```



ගැලීම් සටහන – If-Else-EndIf: ඉරට්ටේ හෝ ඔත්තේ

If-Else-EndIf: ඉරට්ටේ හෝ ඔත්තේ

Pascal

```
program EvenOdd;
var
  n: integer;
begin
  write('Enter a number: ');
  readln(n);
  if n mod 2 = 0 then
    writeln(n, ' is even')
  else
    writeln(n, ' is odd');
end.
```

Python

```
n = int(input("Enter a number: "))
if n % 2 == 0:
    print(n, "is even")
else:
    print(n, "is odd")
```

නියැදි ධාවනය 1

```
Enter a number: 14
14 is even
```

නියැදි ධාවනය 2

```
Enter a number: 7
7 is odd
```

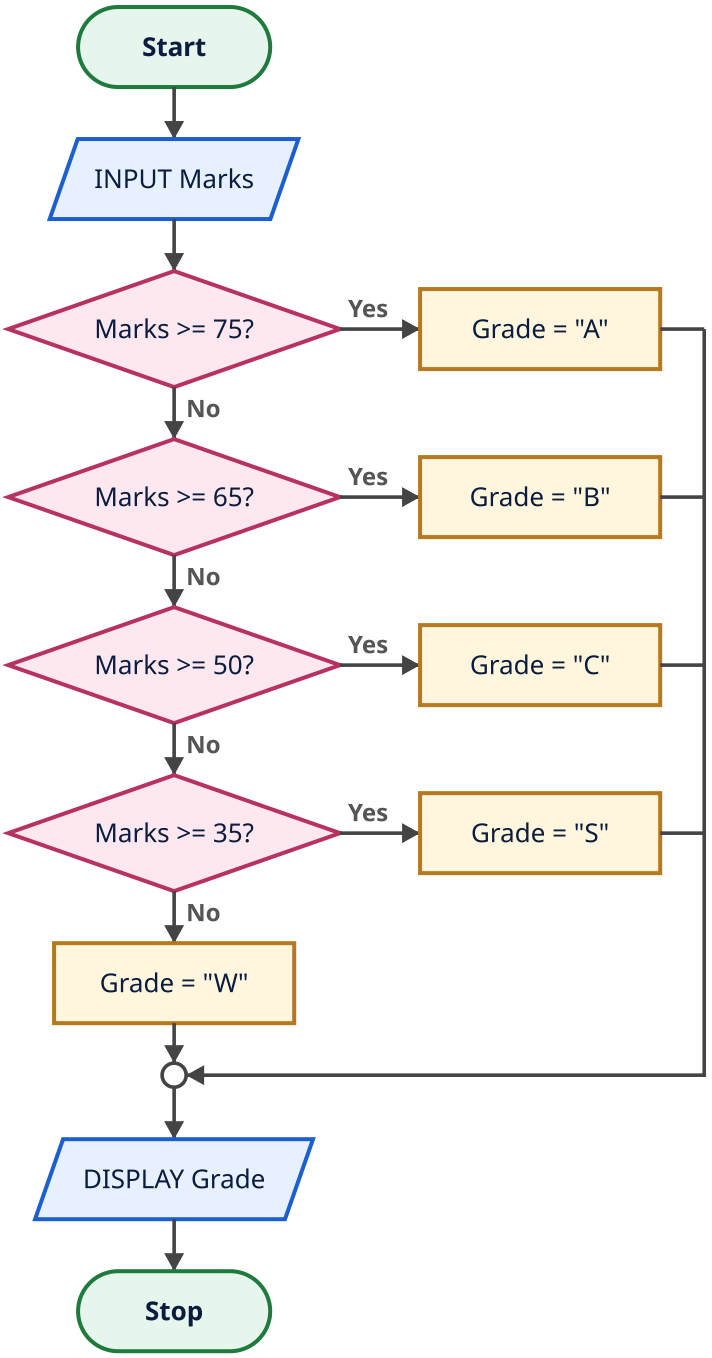
> 💡 Pascal උගුල: **else ට පෙර කිසි විටෙක semicolon දමන්න එපා**. writeln('even') else … ලෙස ලියන්න, writeln('even'); else … ලෙස නොවේ – semicolon එක මුළු if ප්‍රකාශයම අවසන් කරන නිසා සම්පාදකය දෝෂයක් දෙයි. Python ඒ වෙනුවට colon : සහ **ඉන්ඩෙන්ටේෂන්** (හිස්තැන් 4) භාවිත කරයි.

3. විකල්ප බොහොමයක්: IF … ELSE IF (O/L ශ්‍රේණි)

මාර්ග දෙකකට වඩා ඇති විට, කොන්දේසි **එකින් එක** පරීක්ෂා කරමු. පළමු සත්‍ය කොන්දේසිය ජය ගන්නා අතර ඉතිරිය මඟ හැරේ. ශ්‍රී ලංකා O/L ශ්‍රේණි: A ≥ 75, B ≥ 65, C ≥ 50, S ≥ 35, නැතිනම් W.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT Marks
  IF Marks >= 75 THEN
    Grade = "A"
  ELSE IF Marks >= 65 THEN
    Grade = "B"
  ELSE IF Marks >= 50 THEN
    Grade = "C"
  ELSE IF Marks >= 35 THEN
    Grade = "S"
  ELSE
    Grade = "W"
  ENDIF
  DISPLAY Grade
END
```



ගැලීම් සටහන – ලකුණු ශ්‍රේණියට (A, B, C, S, W)

ලකුණු ශ්‍රේණියට (A, B, C, S, W)

Pascal

```
program MarksToGrade;
var
  marks: integer;
  grade: char;
begin
  write('Enter marks: ');
  readln(marks);
  if marks >= 75 then
    grade := 'A'
  else if marks >= 65 then
    grade := 'B'
  else if marks >= 50 then
    grade := 'C'
  else if marks >= 35 then
    grade := 'S'
  else
    grade := 'W';
  writeln('Grade: ', grade);
end.
```

Python

```
marks = int(input("Enter marks: "))
if marks >= 75:
    grade = "A"
elif marks >= 65:
    grade = "B"
elif marks >= 50:
    grade = "C"
elif marks >= 35:
    grade = "S"
else:
    grade = "W"
print("Grade:", grade)
```

නියැදි ධාවනය 1

```
Enter marks: 82
Grade: A
```

නියැදි ධාවනය 2

```
Enter marks: 58
Grade: C
```

නියැදි ධාවනය 3

```
Enter marks: 30
Grade: W
```

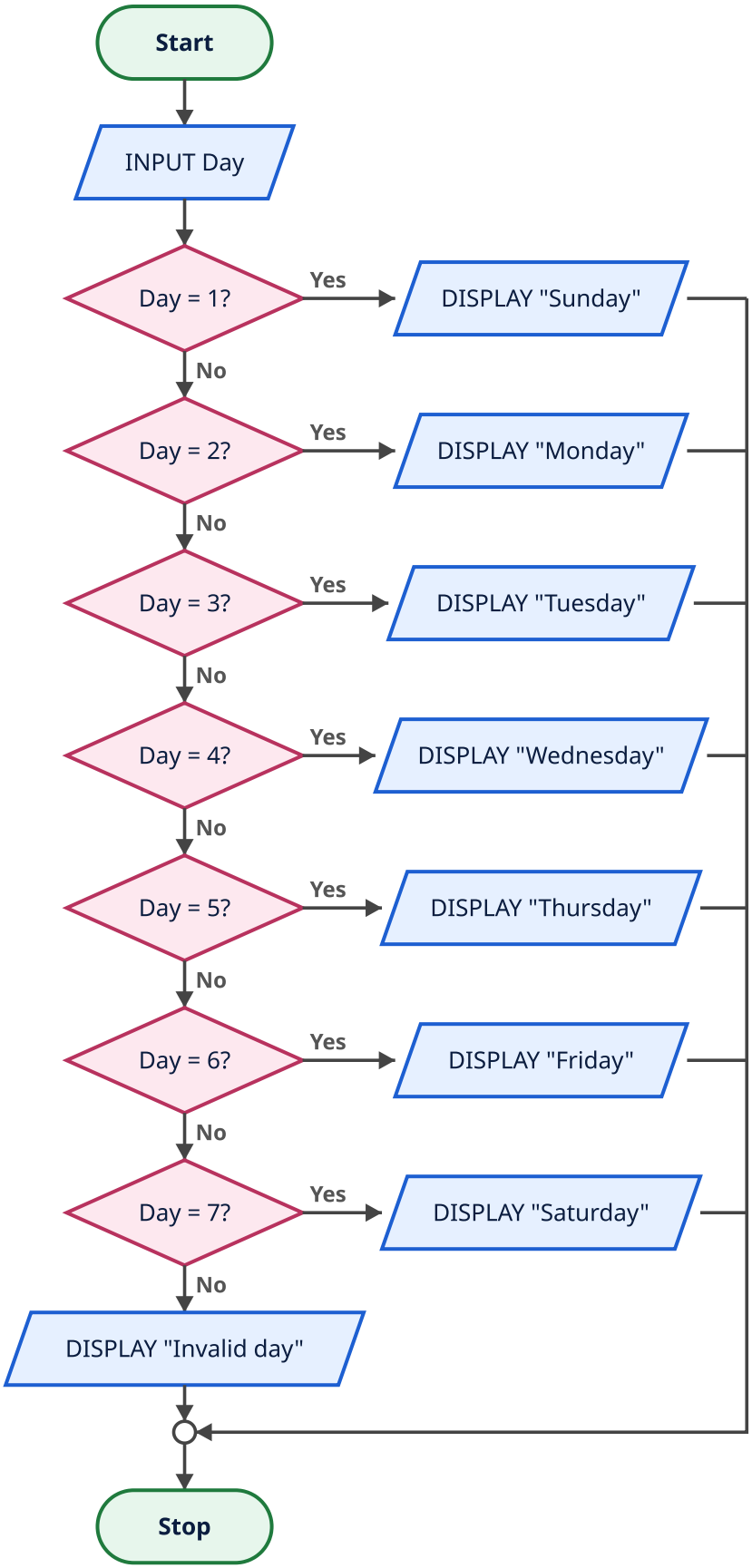
4. Case (බහු-මාර්ග වරණය)

එක් විචල්‍යයක් **ස්ථාවර අගයන්** (1, 2, 3…) බොහොමයක් සමඟ සසඳන විට, IF බොහොමයකට වඩා **Case** කෙටි සහ පැහැදිලිය. Pascal **case … of … else … end** භාවිත කරයි. Python (3.10

සහ පසු) **match … case** භාවිත කරයි; case _ යනු “වෙනත් ඕනෑම දෙයක්”.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT Day
  CASE Day OF
    1 : DISPLAY "Sunday"
    2 : DISPLAY "Monday"
    3 : DISPLAY "Tuesday"
    4 : DISPLAY "Wednesday"
    5 : DISPLAY "Thursday"
    6 : DISPLAY "Friday"
    7 : DISPLAY "Saturday"
    OTHERWISE : DISPLAY "Invalid day"
  ENDCASE
END
```



ගැලීම් සටහන – Case / switch: දින අංකය දින නාමයට

Case / switch: දින අංකය දින නාමයට

Pascal

```
program DayName;
var
  day: integer;
begin
  write('Enter day number (1-7): ');
  readln(day);
  case day of
    1: writeln('Sunday');
    2: writeln('Monday');
    3: writeln('Tuesday');
    4: writeln('Wednesday');
    5: writeln('Thursday');
    6: writeln('Friday');
    7: writeln('Saturday');
  else
    writeln('Invalid day');
  end;
end.
```

Python

```
day = int(input("Enter day number (1-7): "))
match day:
    case 1:
        print("Sunday")
    case 2:
        print("Monday")
    case 3:
        print("Tuesday")
    case 4:
        print("Wednesday")
    case 5:
        print("Thursday")
    case 6:
        print("Friday")
    case 7:
        print("Saturday")
    case _:
        print("Invalid day")
```

නියැදි ධාවනය 1

```
Enter day number (1-7): 3
Tuesday
```

නියැදි ධාවනය 2

```
Enter day number (1-7): 9
Invalid day
```

5. ගැලීම් සටහනේ සිට කේතය දක්වා – පියවර

Its question becomes the IF condition.
එහි ප්‍රශ්නය IF කොන්දේසිය වේ.

Write it after THEN.
THEN ට පසුව ලියන්න.

Write it after ELSE (skip ELSE if empty).
ELSE ට පසුව ලියන්න (හිස් නම් ELSE අවශ්‍ය නැත).

Write ENDIF (Pascal: end of statement; Python: stop indenting).
ENDIF ලියන්න (Pascal: ප්‍රකාශය අවසන්; Python: ඉන්ඩෙන්ට් කිරීම නවත්වන්න).

Use one input for Yes and one for No.
ඔව් සඳහා එක් ආදානයක් සහ නැත සඳහා එකක් භාවිත කරන්න.

ප්‍රධාන පද

වරණය – කොන්දේසියක් භාවිතයෙන් ක්‍රියාත්මක වන ප්‍රකාශ තේරීම.

If-EndIf – කොන්දේසිය සත්‍ය වූ විට පමණක් ප්‍රකාශ ක්‍රියාත්මක කරයි.

If-Else-EndIf – ප්‍රකාශ කණ්ඩායම් දෙකකින් එකක් තෝරයි.

Case – ස්ථාවර අගයන් භාවිතයෙන් විකල්ප බොහොමයකින් එකක් තෝරයි.

THEN – කොන්දේසිය සත්‍ය වූ විට ක්‍රියාත්මක වන කොටස ආරම්භ කරයි.

ELSE – කොන්දේසිය අසත්‍ය වූ විට ක්‍රියාත්මක වන කොටස ආරම්භ කරයි.

ENDIF – ව්‍යාජ කේතයේ IF එකක අවසානය සලකුණු කරයි.

match … case – Python හි case ප්‍රකාශය (Python 3.10+).

ඉන්ඩෙන්ටේෂන් – පේළියක මුල හිස්තැන්; Python ප්‍රකාශ කාණ්ඩ කිරීමට භාවිත කරයි.

ශාඛාව – තීරණයකින් පිටවන මාර්ගවලින් එකක්.

ඉරට්ටේ සංඛ්‍යාව – N mod 2 = 0 වන සංඛ්‍යාවක්.

පරීක්ෂණ දත්ත – වැඩසටහනක සෑම මාර්ගයක්ම පරීක්ෂා කිරීමට තෝරාගත් ආදාන.

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. If-EndIf සහ If-Else-EndIf අතර වෙනස කුමක්ද?

පිළිතුර: If-EndIf යමක් කරයි හෝ කිසිවක් නොකරයි; If-Else-EndIf සැමවිටම දෙකින් එකක් කරයි.

2. රු. 6500 බිලක් සඳහා වැඩසටහන දෙන වට්ටම කීයද?

පිළිතුර: රු. 650; ගෙවිය යුත්තේ රු. 5850.

3. 58 ට ලැබෙන ශ්‍රේණිය කුමක්ද?

පිළිතුර: C (≥ 50 නමුත් < 65).

4. IF බොහොමයකට වඩා Case හොඳ වන්නේ කවදාද?

> **පිළිතුර:** එක් විචල්‍යයක් ස්ථාවර අගයන් බොහොමයක් සමඟ සසඳන විට.

5. day වැඩසටහන 9 සඳහා පෙන්වන්නේ කුමක්ද?

> **පිළිතුර:** Invalid day.

6. IF එකක මාර්ග දෙකම පරීක්ෂා කළ යුත්තේ ඇයි?

> **පිළිතුර:** අප පරීක්ෂා නොකළ මාර්ගයේ වැරැද්දක් සැඟවී තිබිය හැක.

බහුවරණ ප්‍රශ්න

1. If-EndIf හි කොන්දේසිය අසත්‍ය වූ විට...

A. දෝෂයක් ඇතිවේ
B. වැඩසටහන නවතී
C. ප්‍රකාශ දෙවරක් ධාවනය වේ
D. ප්‍රකාශ මඟ හැරේ

2. If-Else-EndIf සැමවිටම ධාවනය කරන්නේ...

A. ELSE කොටස පමණි
B. කොටස් දෙකෙන් හරියටම එකක්
C. කොටස් දෙකම
D. කිසිදු කොටසක් නැත

3. රු. 5000 ට වැඩි නම් 10% වට්ටම සමඟ රු. 6500 බිල. ගෙවිය යුත්තේ...

A. Rs. 650
B. Rs. 5000
C. Rs. 5850
D. Rs. 6500

4. එම වැඩසටහනේම රු. 3000 බිල. වට්ටම...

A. Rs. 0
B. Rs. 3000
C. Rs. 300
D. Rs. 30

5. even/odd වැඩසටහන “7 is odd” පෙන්වන්නේ...

A. 7 > 2 නිසා
B. 7 mod 2 = 1 නිසා
C. 7 mod 2 = 0 නිසා
D. 7 div 2 = 3 නිසා

6. ලකුණු 82 සඳහා O/L ශ්‍රේණිය...

A. S
B. C
C. A
D. B

7. ලකුණු 30 සඳහා O/L ශ්‍රේණිය...

A. W
B. C
C. F
D. S

8. ලකුණු 66 සඳහා O/L ශ්‍රේණිය...

A. A
B. B
C. C
D. S

9. දින අංකය 1–7 න් දින නාමය තේරීමට වඩාත් සුදුසු...

A. Case
B. Boolean අරාවක්
C. FOR ලූපයක්
D. නියතයක්

10. Python හි case ප්‍රකාශය භාවිත කරන වචන...

A. select ... when
B. switch ... case
C. match ... case
D. case ... of

11. Pascal හි else ට කෙළින්ම පෙර නොතිබිය යුත්තේ කුමක්ද?

A. නව පේළියක්
B. හිස්තැනක්
C. writeln
D. semicolon ;

12. Python if එකක් තුළ ප්‍රකාශ කාණ්ඩ කරන්නේ...

A. { } වරහන් මගින්
B. semicolons මගින්
C. ඉන්ඩෙන්ටේෂන් මගින්
D. begin ... end මගින්

13. ගැලීම් සටහනක තීරණයක ඔව් ශාඛාව වන්නේ ... කොටසයි.

A. ELSE
B. THEN
C. CASE
D. ENDIF

14. Pascal case හි “වෙනත් ඕනෑම අගයක්” සඳහා කොටස ලියන්නේ...

A. otherwise only in Python
B. default
C. case _
D. else

15. pass/fail වැඩසටහනක් සම්පූර්ණයෙන් පරීක්ෂා කිරීමට භාවිත කරන්න...

A. සමත් ලකුණක් සහ අසමත් ලකුණක්
B. එකම ලකුණ දෙවරක්
C. සමත් ලකුණක් පමණි
D. ආදානයක් නැත

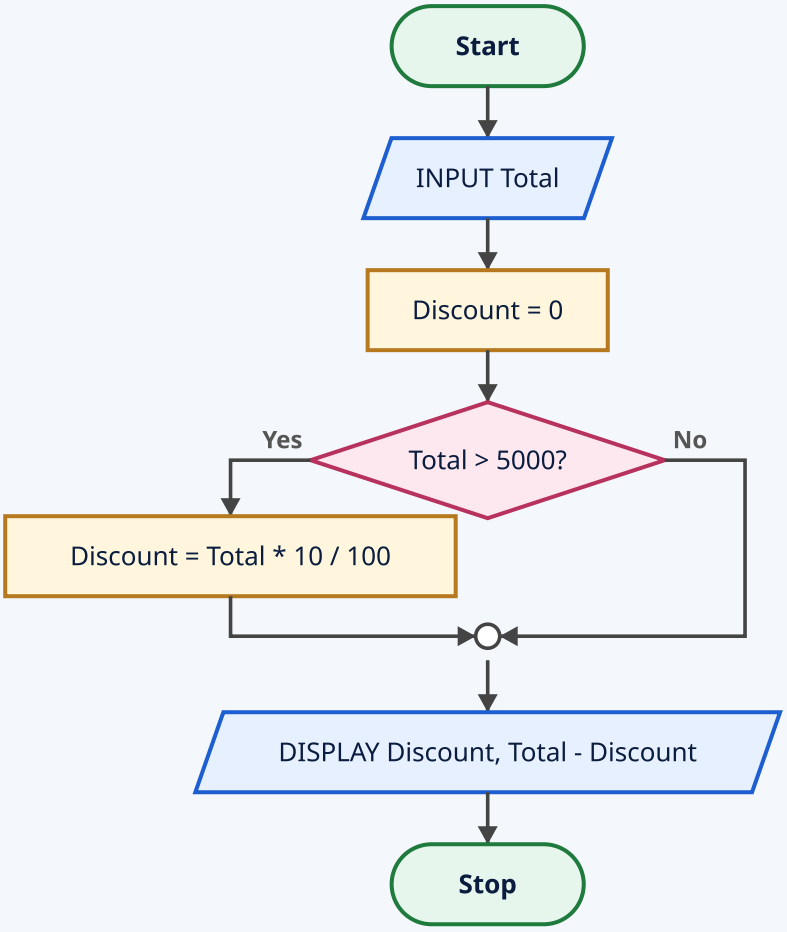
ව්‍යුහගත ප්‍රශ්න

1. බිල රු. 5000 ට වැඩි විට කඩයක් 10% වට්ටමක් දෙයි. (i) ගැලීම් සටහන අඳින්න. (ii) ව්‍යාජ කේතය ලියන්න. (iii) රු. 6500 සහ රු. 3000 සඳහා ගෙවිය යුතු මුදල සොයන්න. (ලකුණු 6) *(ලකුණු 6)*

(i) Start, Total ආදානය, Discount = 0, දියමන්තිය Total > 5000?, ඔව් → Discount = Total × 10 / 100, පෙන්වීම, Stop (2). (ii) පෙන්වා ඇති පරිදි (2). (iii) රු. 5850 සහ රු. 3000 (2).

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT Total
  Discount = 0
  IF Total > 5000 THEN
    Discount = Total * 10 / 100
  ENDIF
  DISPLAY Discount, Total - Discount
END
```



ගැලීම් සටහන – If–EndIf: Rs. 5000 ට වැඩි බිලකට 10% වට්ටම

2. සංඛ්‍යාවක් ආදානය කර එය ඉරට්ටේද ඔත්තේද යන්න පෙන්වන වැඩසටහනක් (Pascal හෝ Python) ලියන්න. (ලකුණු 4) *(ලකුණු 4)*

ආදානය (1); කොන්දේසිය N mod 2 = 0 (1); ඉරට්ටේ පෙන්වීම (1); නැතිනම් ඔත්තේ පෙන්වීම (1).

If–Else–EndIf: ඉරට්ටේ හෝ ඔත්තේ

Pascal

```
program EvenOdd;
var
  n: integer;
begin
  write('Enter a number: ');
  readln(n);
  if n mod 2 = 0 then
    writeln(n, ' is even')
  else
    writeln(n, ' is odd');
end.
```

Python

```
n = int(input("Enter a number: "))
if n % 2 == 0:
    print(n, "is even")
else:
    print(n, "is odd")
```

නියැදි ධාවනය 1

```
Enter a number: 14
14 is even
```

නියැදි ධාවනය 2

```
Enter a number: 7
7 is odd
```

3. ලකුණු ආදානය කර O/L ශ්‍රේණිය පෙන්වන ව්‍යාජ කේතය ලියන්න (A ≥ 75, B ≥ 65, C ≥ 50, S ≥ 35, නැතිනම් W). (ලකුණු 5) *(ලකුණු 5)*

ආදානය (1); IF ≥ 75 A (1); ELSE IF ≥ 65 B, ≥ 50 C (1); ≥ 35 S (1); ELSE W සහ පෙන්වීම (1).

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT Marks
  IF Marks >= 75 THEN
    Grade = "A"
  ELSE IF Marks >= 65 THEN
    Grade = "B"
  ELSE IF Marks >= 50 THEN
    Grade = "C"
  ELSE IF Marks >= 35 THEN
    Grade = "S"
  ELSE
    Grade = "W"
  ENDIF
  DISPLAY Grade
END
```

4. (i) Case ප්‍රකාශයක් භාවිත කරන්නේ කවදාද? (ලකුණු 1)
(ii) 1 සඳහා “Sunday”, 2 සඳහා “Monday” සහ නැතිනම් “Invalid day” පෙන්වන Case ප්‍රකාශයක් ව්‍යාජ කේතයෙන් ලියන්න. (ලකුණු 3) *(ලකුණු 4)*

(i) එක් විචල්‍යයක් ස්ථාවර අගයන් බොහොමයක් සමඟ සසඳන විට. (ii) CASE Day OF 1 : DISPLAY “Sunday”; 2 : DISPLAY “Monday”; OTHERWISE : DISPLAY “Invalid day”; ENDCASE.

5. මෙම Pascal කේතයේ දෝෂය සොයා නිවැරදි කරන්න: if Mark >= 35 then writeln('Pass'); else writeln('Fail'); (ලකුණු 2) *(ලකුණු 2)*

else ට පෙර semicolon නොතිබිය යුතුය (1). නිවැරදි: if Mark >= 35 then writeln('Pass') else writeln('Fail'); (1).

සාරාංශය

- වරණය = තීරණයක් ගැනීම.
- If–EndIf: කරන්න හෝ මඟ හරින්න.
- If–Else–EndIf: මාර්ග දෙකෙන් එකක්.
- ELSE IF දාම කොන්දේසි පිළිවෙළින් පරීක්ෂා කරයි.
- Case: ස්ථාවර අගයන් බොහොමයකින් එකක්.
- Pascal: case … of; Python: match … case.
- Pascal: else ට පෙර ; නැත. Python: ඉන්ඩෙන්ට්.
- පරීක්ෂණ දත්ත මගින් සෑම මාර්ගයක්ම පරීක්ෂා කරන්න.

MCQ පිළිතුරු – 1.6

1. **D** – මඟ හැරේ.
2. **B** – දෙකෙන් එකක්.
3. **C** – 6500 − 650.
4. **A** – 5000 ට වැඩි නැත.
5. **B** – ශේෂය 1.
6. **C** – ≥ 75.
7. **A** – < 35.
8. **B** – 65–74.
9. **A** – ස්ථාවර අගයන් බොහොමයක්.
10. **C** – Python 3.10+.
11. **D** – Semicolon if අවසන් කරයි.
12. **C** – ඉන්ඩෙන්ටේෂන්.
13. **B** – ඔව් → THEN.
14. **D** – Pascal: else.
15. **A** – මාර්ග දෙකම පරීක්ෂා කරන්න.

© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.

G.C.E. O/L ICT · 11 ශ්‍රේණිය · පාඩම 1

1.7 පුනරාවර්තනය: FOR, WHILE සහ REPEAT-UNTIL ලූප

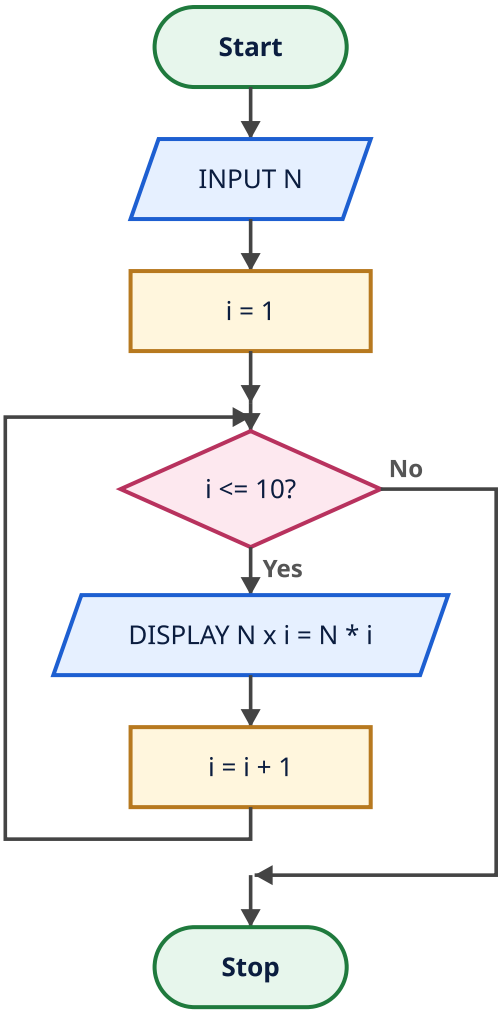
විෂය නිර්දේශය 10.7. ලූපයක් ප්‍රකාශ නැවත නැවත ක්‍රියාත්මක කරයි. **කී වරක්** කළ යුතුදැයි **දන්නේ නම්** (උදා 10 වරක්) **FOR** ලූපයක් භාවිත කරමු. **නොදන්නේ නම්** (උදා “පරිශීලකයා −1 ටයිප් කරන තුරු”) **ආරම්භයේදී කොන්දේසිය පරීක්ෂා කරන WHILE** ලූපයක් හෝ **අවසානයේදී පරීක්ෂා කරන REPEAT-UNTIL** ලූපයක් භාවිත කරමු. එක් ලූපයක් තවත් එකකට **පරිවර්තනය** කිරීමද ඉගෙන ගනිමු.

1. දන්නා වාර ගණන: FOR

FOR ලූපයක් ආරම්භක අගයක සිට අවසාන අගයක් දක්වා යන **ගණකයක්** (counter) භාවිත කරයි. සෑම අගයකටම ලූපයේ ඇතුළත එක් වරක් ධාවනය වේ. උදාහරණය: සංඛ්‍යාවක ගුණන වගුව, 1 සිට 10 දක්වා.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT N
  FOR i = 1 TO 10
    DISPLAY N, " x ", i, " = ", N * i
  ENDFOR
END
```



ගැලීම් සටහන – For loop (දන්නා වාර ගණන): ගුණන වගුව

For loop (දන්නා වාර ගණන): ගුණන වගුව

Pascal

```
program TimesTable;
var
  n, i: integer;
begin
  write('Enter a number: ');
  readln(n);
  for i := 1 to 10 do
    writeln(n, ' x ', i, ' = ', n * i);
end.
```

Python

```
n = int(input("Enter a number: "))
for i in range(1, 11):
    print(n, "x", i, "=", n * i)
```

නියැදි ධාවනය

```
Enter a number: 7
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
```

FOR ලූපයක අනුරේඛනය – 5 හි සාධකීය (F = 1 × 2 × 3 × 4 × 5):

FOR loop මගින් ක්‍රමාරෝපිතය (factorial)

Pascal

```
program Factorial;
var
  n, i, f: integer;
begin
  write('Enter N: ');
  readln(n);
  f := 1;
  for i := 1 to n do
    f := f * i;
  writeln('Factorial = ', f);
end.
```

Python

```
n = int(input("Enter N: "))
f = 1
for i in range(1, n + 1):
    f = f * i
print("Factorial =", f)
```

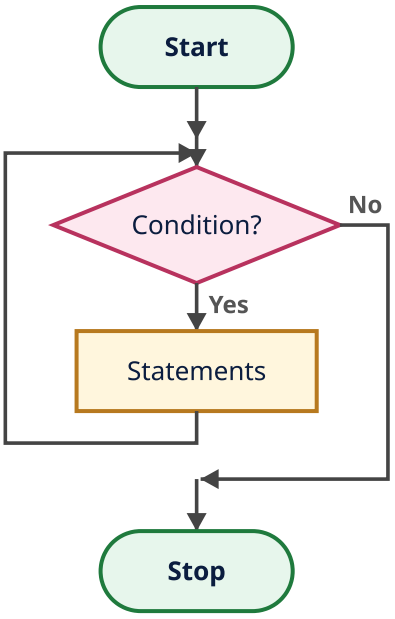
නියැදි ධාවනය

```
Enter N: 5
Factorial = 120
```

i	F = F * i
1	1
2	2
3	6
4	24
5	120

2. නොදන්නා වාර ගණන, ආරම්භයේ පරීක්ෂාව: WHILE

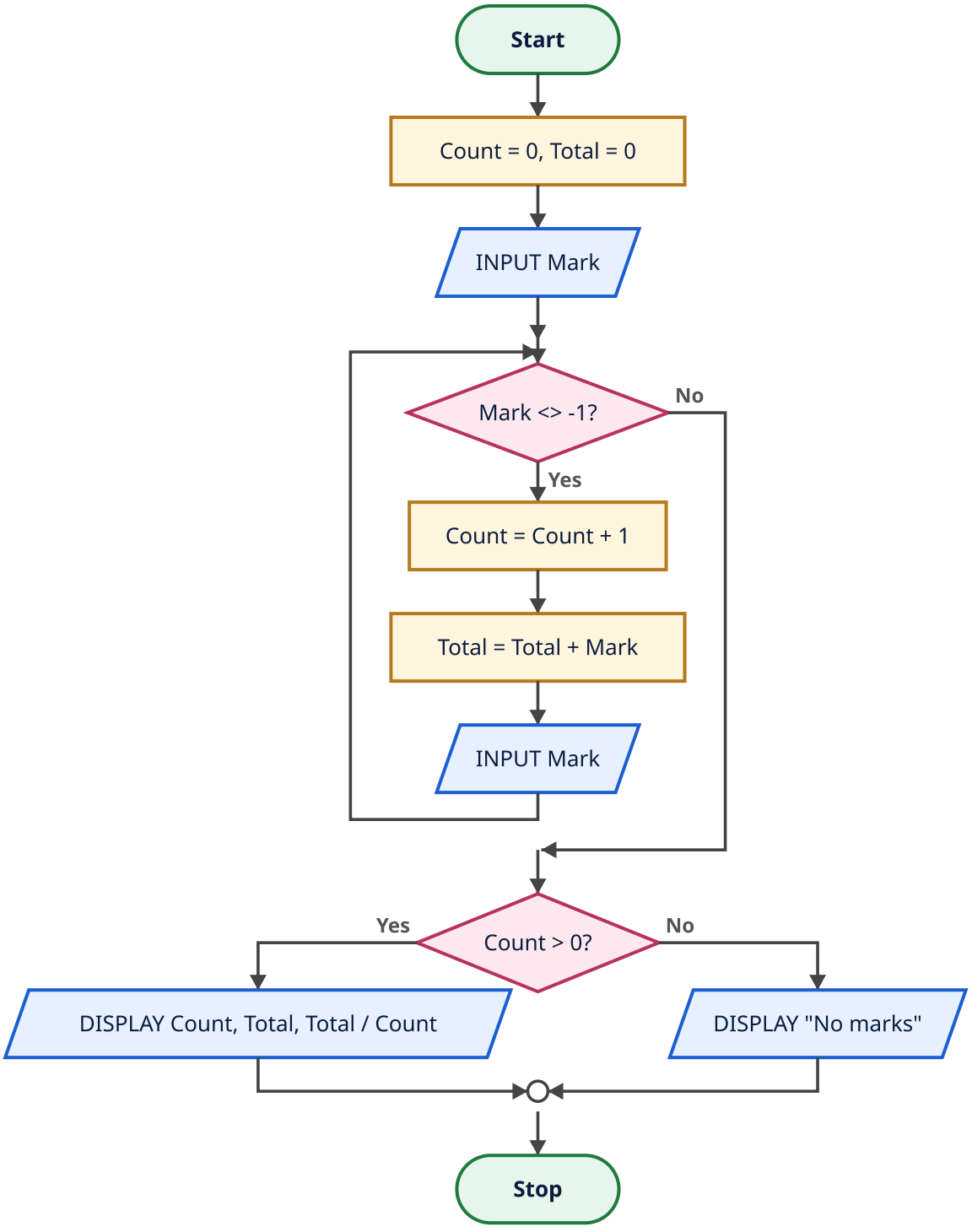
WHILE ලූපයක් කොන්දේසිය **මුලින්ම** පරීක්ෂා කරයි. සත්‍ය නම් ඇතුළත ධාවනය වී කොන්දේසිය නැවත පරීක්ෂා කරයි. ආරම්භයේදීම අසත්‍ය නම් ඇතුළත **ශූන්‍ය වරක්** ධාවනය වේ. උදාහරණය: −1 ටයිප් කරන තුරු ලකුණු ඇතුළත් කිරීම. −1 යනු ලකුණක් නොව **sentinel** (නැවතීමේ සංඥාවක්) අගයකි.



WHILE ලූපයක සාමාන්‍ය ආකාරය (ආරම්භයේ පරීක්ෂාව)

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  Count = 0
  Total = 0
  INPUT Mark
  WHILE Mark <> -1 DO
    Count = Count + 1
    Total = Total + Mark
    INPUT Mark
  ENDWHILE
  IF Count > 0 THEN
    DISPLAY Count, Total, Total / Count
  ELSE
    DISPLAY "No marks entered"
  ENDIF
END
```



ගැලීම් සටහන – While loop (නොදන්නා වාර ගණන): −1 දක්වා ලකුණු

While loop (නොදන්නා වාර ගණන): −1 දක්වා ලකුණු

Pascal

```
program MarksUntilStop;
var
  mark, count, total: integer;
begin
  count := 0;
  total := 0;
  write('Enter mark (-1 to stop): ');
  readln(mark);
  while mark <> -1 do
  begin
    count := count + 1;
    total := total + mark;
    write('Enter mark (-1 to stop): ');
    readln(mark);
  end;
  if count > 0 then
  begin
    writeln('Count = ', count);
    writeln('Total = ', total);
    writeln('Average = ', total / count:0:2);
  end
  else
    writeln('No marks entered');
end.
```

Python

```
count = 0
total = 0
mark = int(input("Enter mark (-1 to stop): "))
while mark != -1:
    count = count + 1
    total = total + mark
    mark = int(input("Enter mark (-1 to stop): "))
if count > 0:
    print("Count =", count)
    print("Total =", total)
    print(f"Average = {total / count:.2f}")
else:
    print("No marks entered")
```

නියැදි ධාවනය 1

```
Enter mark (-1 to stop): 70
Enter mark (-1 to stop): 85
Enter mark (-1 to stop): 64
Enter mark (-1 to stop): -1
Count = 3
Total = 219
Average = 73.00
```

නියැදි ධාවනය 2

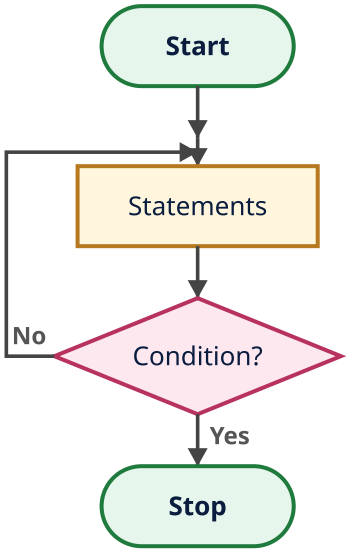
```
Enter mark (-1 to stop): -1
No marks entered
```

Mark	Count	Total
70	1	70
85	2	155
64	3	219
-1	3	219

Count = 3, Total = 219, එබැවින් සාමාන්‍යය = 219 ÷ 3 = 73.00. පළමු ආදානය −1 නම් ලූපයේ ඇතුළත කිසි විටෙක ධාවනය නොවී “No marks entered” කියයි.

3. නොදන්නා වාර ගණන, අවසානයේ පරීක්ෂාව: REPEAT–UNTIL

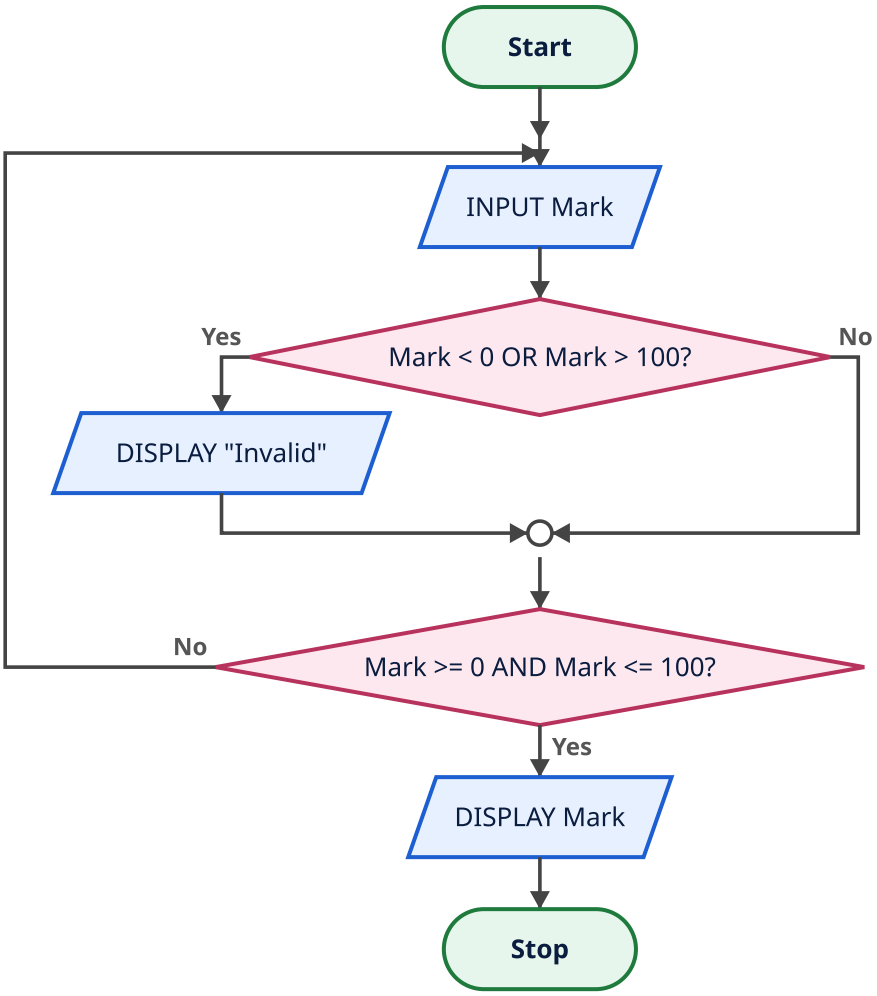
REPEAT–UNTIL ලූපයක් ඇතුළත **මුලින්** ධාවනය කර **අවසානයේදී** කොන්දේසිය පරීක්ෂා කරයි, එබැවින් ඇතුළත සැමවිටම **අවම වශයෙන් එක් වරක්** ධාවනය වේ. කොන්දේසිය **සත්‍ය** වූ විට නවතී. ආදාන පරීක්ෂාවට ඉතා හොඳයි: වලංගු ලකුණක් (0–100) ටයිප් කරන තුරු දිගටම ඉල්ලීම.



REPEAT–UNTIL ලූපයක සාමාන්‍ය ආකාරය (අවසානයේ පරීක්ෂාව)

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  REPEAT
    INPUT Mark
    IF Mark < 0 OR Mark > 100 THEN
      DISPLAY "Invalid. Try again."
    ENDIF
  UNTIL Mark >= 0 AND Mark <= 100
  DISPLAY Mark
END
```



ගැලීම් සටහන – Repeat–Until (අවසානයේ පරීක්ෂාව): වලංගු ලකුණක් ලබාගැනීම

Repeat-Until (අවසානයේ පරීක්ෂාව): වලංගු ලකුණක් ලබාගැනීම

Pascal

```
program ValidMark;
var
  mark: integer;
begin
  repeat
    write('Enter mark (0-100): ');
    readln(mark);
    if (mark < 0) or (mark > 100) then
      writeln('Invalid. Try again.');
  until (mark >= 0) and (mark <= 100);
  writeln('Mark accepted: ', mark);
end.
```

Python

```
while True:
    mark = int(input("Enter mark (0-100): "))
    if 0 <= mark <= 100:
        break
    print("Invalid. Try again.")
print("Mark accepted:", mark)
```

නියැදි ධාවනය

```
Enter mark (0-100): 120
Invalid. Try again.
Enter mark (0-100): -5
Invalid. Try again.
Enter mark (0-100): 76
Mark accepted: 76
```

💡 Python හි **REPEAT-UNTIL නැත**. while True: ... සහ නැවතීමේ කොන්දේසිය ලැබුණු විට break භාවිතයෙන් එය අනුකරණය කරමු.

4. ලූප තුන සැසඳීම

	FOR	WHILE	REPEAT-UNTIL
වාර ගණන	දනී	නොදනී	නොදනී
කොන්දේසිය පරීක්ෂාව	ගණකය ආරම්භයේ	ආරම්භයේ	අවසානයේ
අවම ධාවන	0	0	1
නවතින්නේ	ගණකය අවසාන අගය ඉක්මවූ විට	කොන්දේසිය අසත්‍ය වූ විට	කොන්දේසිය සත්‍ය වූ විට
Pascal	for i := 1 to 10 do	while ... do	repeat ... until ...;
Python	for i in range(1, 11):	while ...:	while True: ... break

5. එක් ලූපයක් තවත් එකකට පරිවර්තනය

එකම කාර්යය – 2, 4, 6, 8, 10 ඉරට්ටේ සංඛ්‍යා පෙන්වීම – එක් එක් ලූපයෙන් ලියා ඇත. FOR එකක් WHILE එකකට වෙනස් කිරීමට: (1) ලූපයට **පෙර** ගණකය සකසන්න, (2) කොන්දේසිය ලියන්න, (3) ලූපය **තුළ ගණකය වැඩි කරන්න**. REPEAT–UNTIL සඳහා කොන්දේසිය **ප්‍රතිවිරුද්ධ** කරන්න (WHILE i <= 10 වන්නේ UNTIL i > 10).

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  FOR i = 1 TO 5
    DISPLAY i * 2
  ENDFOR
END
```

FOR මගින් 2–10 ඉරට්ටේ සංඛ්‍යා

Pascal

```
program EvenFor;
var
  i: integer;
begin
  for i := 1 to 5 do
    writeln(i * 2);
end.
```

Python

```
for i in range(2, 11, 2):
    print(i)
```

නියැදි ධාවනය

```
2
4
6
8
10
```

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  i = 2
  WHILE i <= 10 DO
    DISPLAY i
    i = i + 2
  ENDWHILE
END
```

WHILE මගින් 2–10 ඉරට්ටේ සංඛ්‍යා

Pascal

```
program EvenWhile;
var
  i: integer;
begin
  i := 2;
  while i <= 10 do
  begin
    writeln(i);
    i := i + 2;
  end;
end.
```

Python

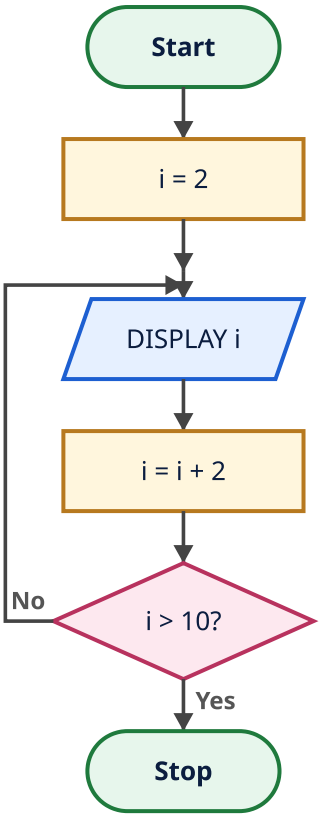
```
i = 2
while i <= 10:
    print(i)
    i = i + 2
```

නියැදි ධාවනය

```
2
4
6
8
10
```

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  i = 2
  REPEAT
    DISPLAY i
    i = i + 2
  UNTIL i > 10
END
```



ගැලීම් සටහන – REPEAT-UNTIL මගින් 2-10 ඉරට්ටේ සංඛ්‍යා

REPEAT-UNTIL මගින් 2-10 ඉරට්ටේ සංඛ්‍යා

Pascal

```
program EvenRepeat;
var
  i: integer;
begin
  i := 2;
  repeat
    writeln(i);
    i := i + 2;
  until i > 10;
end.
```

Python

```
i = 2
while True:          # Python has no repeat-until
    print(i)
    i = i + 2
    if i > 10:
        break
```

නියැදි ධාවනය

```
2
4
6
8
10
```

💡 අනතුර – **අනන්ත ලූපය**: WHILE ලූපයක් තුළ i = i + 2 අමතක වුවහොත් i සදහටම 2 ලෙස පවතින අතර ලූපය කිසි විටෙක නොනවතී. ලූපය තුළ යමක් එය අවසානය දෙසට ගෙන යන බව සැමවිටම පරීක්ෂා කරන්න.

ප්‍රධාන පද

ලූපය – නැවත නැවත ක්‍රියාත්මක වන ප්‍රකාශ.

FOR ලූපය – ගණකයක් භාවිතයෙන් දන්නා වාර ගණනක් ධාවනය වන ලූපය.

WHILE ලූපය – ආරම්භයේදී කොන්දේසිය පරීක්ෂා කරන ලූපය.

REPEAT–UNTIL ලූපය – අවසානයේදී කොන්දේසිය පරීක්ෂා කරන ලූපය; අවම වශයෙන් එක් වරක් ධාවනය වේ.

ගණකය – ලූප ධාවන ගණන් කරන විචල්‍යයක්, උදා i.

පූර්ව-පරීක්ෂණ ලූපය – ඇතුළතට පෙර කොන්දේසිය පරීක්ෂාව (WHILE, FOR).

පශ්චාත්-පරීක්ෂණ ලූපය – ඇතුළතට පසු කොන්දේසිය පරීක්ෂාව (REPEAT–UNTIL).

Sentinel අගය – “නවතින්න” යන්න අදහස් කරන විශේෂ අගයක්, උදා −1.

අනන්ත ලූපය – කිසි විටෙක නොනවතින ලූපයක්.

වලංගුකරණය – ආදානය සාධාරණදැයි පරීක්ෂාව, උදා 0 සිට 100 දක්වා ලකුණක්.

range() – FOR ලූපයකට සංඛ්‍යා දෙන Python ශ්‍රිතය; range(1, 11) යනු 1 සිට 10.

break – ලූපයකින් වහාම පිටවන Python වචනය.

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. FOR ලූපයක් භාවිත කරන්නේ කවදාද?

පිළිතුර: කී වරක් පුනරාවර්තනය කළ යුතුදැයි දන්නා විට.

2. WHILE සහ REPEAT–UNTIL අතර ප්‍රධාන වෙනස කුමක්ද?

පිළිතුර: WHILE ආරම්භයේ පරීක්ෂා කරයි (0 වරක් ධාවනය විය හැක); REPEAT–UNTIL අවසානයේ පරීක්ෂා කරයි (අවම වශයෙන් එක් වරක්).

3. Sentinel අගයක් යනු කුමක්ද?

පිළිතුර: ලූපයට නවතින ලෙස කියන −1 වැනි විශේෂ අගයක්.

4. 5 හි සාධකීය කුමක්ද?

පිළිතුර: 120.

5. for i := 1 to 10 කී වරක් ධාවනය වේද?

පිළිතුර: 10 වරක්.

6. අනන්ත ලූපයක් ඇති වන්නේ කුමක් නිසාද?

> **පිළිතුර:** කොන්දේසිය කිසි විටෙක වෙනස් නොවීම, උදා ගණකය වැඩි කිරීම අමතක වීම.

බහුවරණ ප්‍රශ්න

1. දන්නා වාර ගණනක් සඳහා ලූපය...

A. FOR
B. WHILE
C. IF
D. REPEAT–UNTIL

2. සැමවිටම අවම වශයෙන් එක් වරක් ධාවනය වන ලූපය කුමක්ද?

A. WHILE
B. FOR
C. කිසිවක් නැත
D. REPEAT–UNTIL

3. ආරම්භයේදී කොන්දේසිය පරීක්ෂා කරන ලූපය කුමක්ද?

A. CASE
B. WHILE
C. ENDIF
D. REPEAT–UNTIL

4. REPEAT ... UNTIL කොන්දේසිය නවතින්නේ කොන්දේසිය...

A. ශූන්‍ය වූ විට
B. අසත්‍ය වූ විට
C. සත්‍ය වූ විට
D. 10 ට සමාන වූ විට

5. Python හි range(1, 11) දෙන්නේ...

A. 1 සිට 10
B. 0 සිට 10
C. 1 සිට 11
D. 0 සිට 11

6. N = 5 සඳහා සාධකීය වැඩසටහනේ ප්‍රතිදානය...

A. Factorial = 60
B. Factorial = 120
C. Factorial = 15
D. Factorial = 25

7. ලකුණු 70, 85, 64 පසුව −1 ඇතුළත් කරයි. පෙන්වන සාමාන්‍යය...

A. 54.50
B. 72.67
C. 73.00
D. 219.00

8. ලකුණු වැඩසටහනේ −1 හඳුන්වන්නේ...

A. දත්ත වර්ගයක්
B. ගණකයක්
C. නියත ලකුණක්
D. sentinel අගයක්

9. පළමු ආදානය −1 නම් WHILE ලූපයේ ඇතුළත ධාවනය වන්නේ...

A. 0 වරක්
B. සදහටම
C. 3 වරක්
D. 1 වරක්

10. Python හි REPEAT–UNTIL නැත. අප භාවිත කරන්නේ...

A. while True: … break
B. repeat()
C. do … loop
D. for … until

11. “WHILE i <= 10” REPEAT–UNTIL ට වෙනස් කළ විට...

A. UNTIL i < 10
B. UNTIL i <= 10
C. UNTIL i > 10
D. UNTIL i = 2

12. WHILE ලූපයක් තුළ ගණකය වැඩි කිරීම අමතක වීමෙන් ඇති වන්නේ...

A. syntax දෝෂයක්
B. ප්‍රතිදානයක් නැත
C. අනන්ත ලූපයක්
D. වේගවත් වැඩසටහනක්

13. ගුණන වගු වැඩසටහන ආදානයෙන් පසු පේළි කීයක් මුද්‍රණය කරයි?

A. 9
B. 10
C. 11
D. 7

14. වලංගු ලකුණක් ටයිප් කරන තුරු නැවත නැවත ඉල්ලීමට වඩාත් සුදුසු...

A. FOR
B. නියතයක්
C. CASE
D. REPEAT–UNTIL

15. for i := 1 to 5 do write(i * 2) මුද්‍රණය කරන්නේ...

A. 1 2 3 4 5
B. 2 4 6 8 10
C. 1 3 5 7 9
D. 2 4 6 8

ව්‍යුහගත ප්‍රශ්න

1. (i) පූර්ව-පරීක්ෂණ සහ පශ්චාත්-පරීක්ෂණ ලූප අතර එක් වෙනසක් දෙන්න. (ලකුණු 2)
(ii) Pascal හි එක් එකකට උදාහරණයක් දෙන්න. (ලකුණු 2) *(ලකුණු 4)*

> (i) පූර්ව-පරීක්ෂණ ලූපය ඇතුළතට පෙර පරීක්ෂා කරන අතර 0 වරක් ධාවනය විය හැක; පශ්චාත්-පරීක්ෂණ ලූපය පසුව පරීක්ෂා කර අවම වශයෙන් එක් වරක් ධාවනය වේ. (ii) while ... do (පූර්ව); repeat ... until (පශ්චාත්).

2. සංඛ්‍යාවක් ආදානය කර එහි ගුණන වගුව 1 සිට 10 දක්වා පෙන්වන වැඩසටහනක් (Pascal හෝ Python) ලියන්න. (ලකුණු 4) *(ලකුණු 4)*

N ආදානය (1); FOR i = 1 සිට 10 (1); N × i ගණනය (1); 7 x 1 = 7 ආකාරයෙන් පෙන්වීම (1).

For loop (දන්නා වාර ගණන): ගුණන වගුව

Pascal

```
program TimesTable;
var
  n, i: integer;
begin
  write('Enter a number: ');
  readln(n);
  for i := 1 to 10 do
    writeln(n, ' x ', i, ' = ', n * i);
end.
```

Python

```
n = int(input("Enter a number: "))
for i in range(1, 11):
    print(n, "x", i, "=", n * i)
```

නියැදි ධාවනය

```
Enter a number: 7
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
```

3. −1 ඇතුළත් කරන තුරු ලකුණු කියවා, පසුව ගණන, එකතුව සහ සාමාන්‍යය පෙන්වන ව්‍යාජ කේතය ලියන්න. (ලකුණු 5) *(ලකුණු 5)*

Count = 0, Total = 0 (1); ලූපයට පෙර Mark ආදානය (1); WHILE Mark <> −1 Count සහ Total ට එකතු කිරීම (1); ලූපය තුළ Mark ආදානය (1); Count, Total, Total / Count පෙන්වීම (1).

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  Count = 0
  Total = 0
  INPUT Mark
  WHILE Mark <> -1 DO
    Count = Count + 1
    Total = Total + Mark
    INPUT Mark
  ENDWHILE
  IF Count > 0 THEN
    DISPLAY Count, Total, Total / Count
  ELSE
    DISPLAY "No marks entered"
  ENDIF
END
```

Mark	Count	Total
70	1	70
85	2	155
64	3	219
-1	3	219

4. මෙම FOR ලූපය WHILE ලූපයක් ලෙස නැවත ලියන්න: FOR i = 1 TO 5 : DISPLAY i * 2 : ENDFOR. (ලකුණු 3) *(ලකුණු 3)*

i = 1 (1); WHILE i <= 5 DO DISPLAY i * 2 (1); i = i + 1; ENDWHILE (1). (නැතහොත් i = 2 … i <= 10 … i = i + 2 සමාන ආකාරය.)

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  i = 2
  WHILE i <= 10 DO
    DISPLAY i
    i = i + 2
  ENDWHILE
END
```

5. N = 5 සමඟ සාධකීය වැඩසටහන සඳහා අනුරේඛන වගුවක් සම්පූර්ණ කර ප්‍රතිදානය දෙන්න. (ලකුණු 3) *(ලකුණු 3)*

i = 1 සිට 5 සඳහා F = 1, 2, 6, 24, 120 (2); ප්‍රතිදානය Factorial = 120 (1).

i	F = F * i
1	1
2	2
3	6
4	24
5	120

සාරාංශය

- FOR: දන්නා වාර ගණන.
- WHILE: ආරම්භයේ පරීක්ෂාව; 0 වරක් ධාවනය විය හැක.
- REPEAT–UNTIL: අවසානයේ පරීක්ෂාව; අවම වශයෙන් එක් වරක්.
- REPEAT නවතින්නේ කොන්දේසිය සත්‍ය වූ විටය.
- Sentinel (−1) ලූපයට නවතින ලෙස කියයි.
- Python: for … in range(); while; while True + break.
- පරිවර්තනය: ගණකය සකසන්න, පරීක්ෂා කරන්න, තුළ වැඩි කරන්න.
- ලූපය අවසානය දෙසට ගෙන යන්න – අනන්ත ලූප වළකින්න.

MCQ පිළිතුරු – 1.7

1. **A** – දන්නා ගණන → FOR.
2. **D** – අවසානයේ පරීක්ෂාව.
3. **B** – පූර්ව-පරීක්ෂණ.
4. **C** – සත්‍ය වන තුරු.
5. **A** – අවසානය ඇතුළත් නැත.
6. **B** – 1×2×3×4×5.
7. **C** – 219 ÷ 3.
8. **D** – නැවතීමේ සංඥාව.
9. **A** – ආරම්භයේදීම අසත්‍යයි.
10. **A** – break සමඟ while True.
11. **C** – කොන්දේසිය ප්‍රතිවිරුද්ධ කරන්න.
12. **C** – නොනවතී.
13. **B** – i = 1 සිට 10.
14. **D** – අවම වශයෙන් එක් වරක් ඉල්ලිය යුතුය.
15. **B** – 1–5 දෙගුණ.

© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.

G.C.E. O/L ICT · 11 ශ්‍රේණිය · පාඩම 1

1.8 කැදැලි පාලන ව්‍යුහ

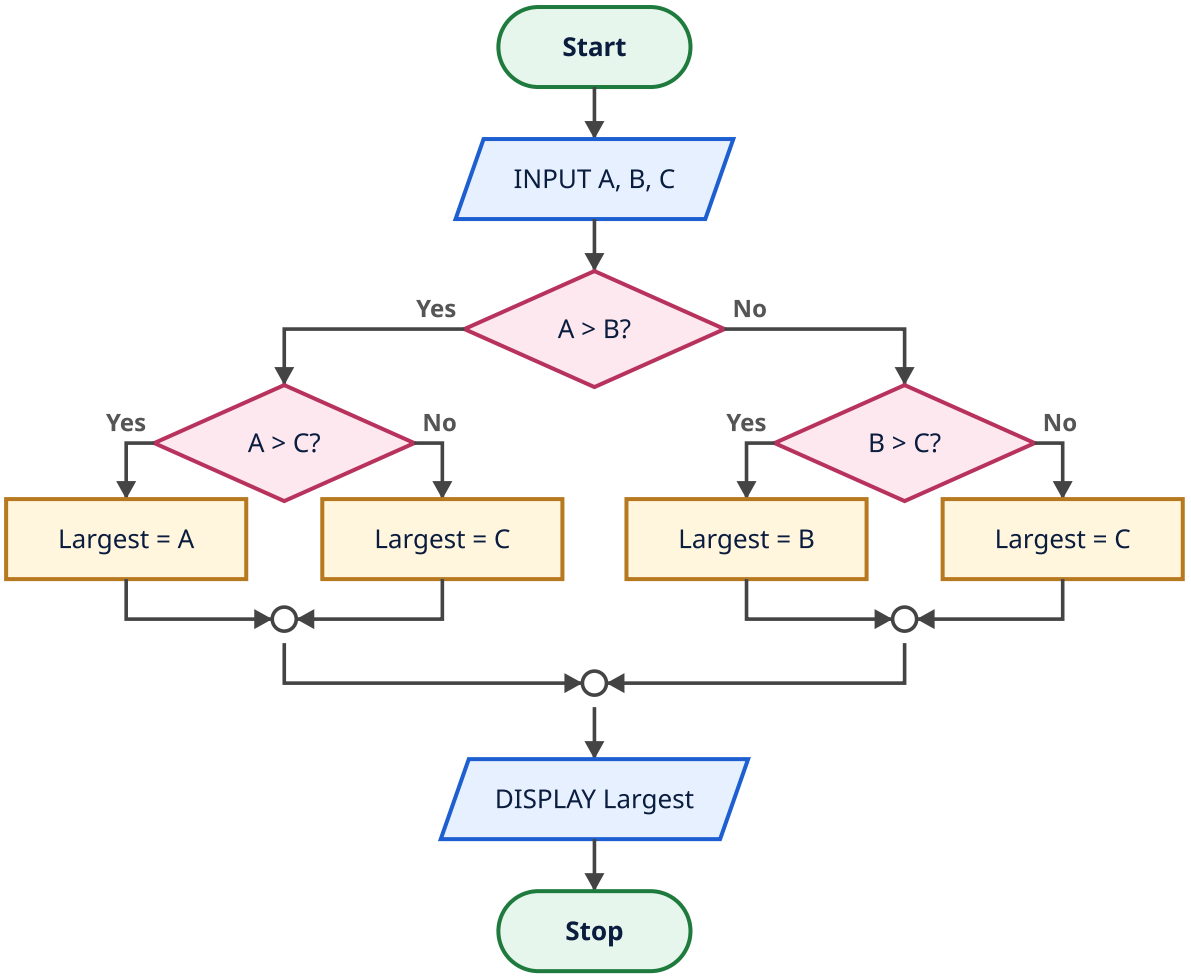
විෂය නිර්දේශය 10.8. වෙනත් ව්‍යුහයක් **තුළ** තැබූ ව්‍යුහයක් **කැදැලි** (nested) වේ – විශාල පෙට්ටියක් තුළ ඇති කුඩා පෙට්ටියක් මෙන්. වරණයක් තුළ වරණයක්, ලූපයක් තුළ ලූපයක්, ලූපයක් තුළ වරණයක් හෝ වරණයක් තුළ ලූපයක් තැබිය හැක. සැබෑ වැඩසටහන් නිතරම කැදැලි භාවිත කරයි.

1. වරණයක් තුළ වරණයක්

A, B සහ C සංඛ්‍යා තුනෙන් විශාලතම එක සොයා ගැනීමට: මුලින් “A > B ද?” යැයි අසන්න. පසුව **එක් එක් පිළිතුර තුළ** C සමඟ දෙවන ප්‍රශ්නයක් අසන්න. ඇතුළු IF එක ධාවනය වන්නේ පිටත IF එක එයට ළඟා වූ විට පමණි.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT A, B, C
  IF A > B THEN
    IF A > C THEN
      Largest = A
    ELSE
      Largest = C
    ENDIF
  ELSE
    IF B > C THEN
      Largest = B
    ELSE
      Largest = C
    ENDIF
  ENDIF
  DISPLAY Largest
END
```



ගැලීම් සටහන – වරණය තුළ වරණය: තුනෙන් විශාලතම

වරණය තුළ වරණය: තුනෙන් විශාලතම

Pascal

```
program Largest3;
var
  a, b, c, largest: integer;
begin
  write('Enter A: ');
  readln(a);
  write('Enter B: ');
  readln(b);
  write('Enter C: ');
  readln(c);
  if a > b then
  begin
    if a > c then
      largest := a
    else
      largest := c;
  end
  else
  begin
    if b > c then
      largest := b
    else
      largest := c;
  end;
  writeln('Largest = ', largest);
end.
```

Python

```
a = int(input("Enter A: "))
b = int(input("Enter B: "))
c = int(input("Enter C: "))
if a > b:
    if a > c:
        largest = a
    else:
        largest = c
else:
    if b > c:
        largest = b
    else:
        largest = c
print("Largest =", largest)
```

නියැදි ධාවනය 1

```
Enter A: 12
Enter B: 45
Enter C: 30
Largest = 45
```

නියැදි ධාවනය 2

```
Enter A: 50
Enter B: 20
```

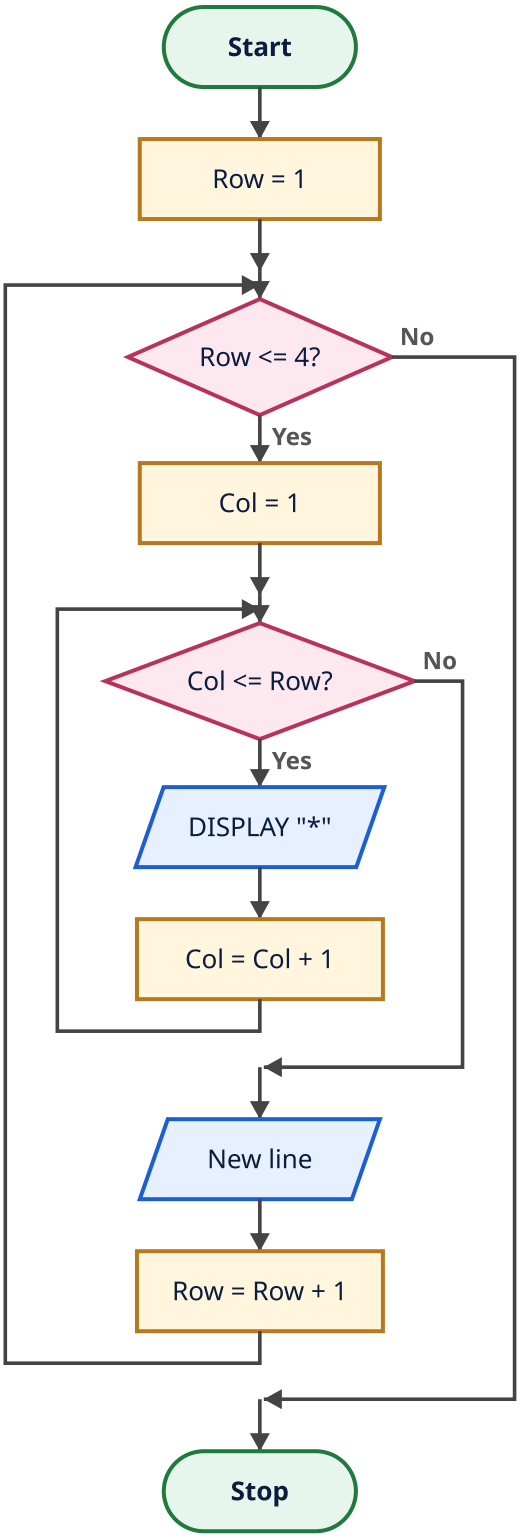
```
Enter C: 35
Largest = 50
```

2. ලූපයක් තුළ ලූපයක්

පිටත ලූපයේ **සෑම ධාවනයකටම ඇතුළු ලූපය සම්පූර්ණයෙන්** ධාවනය වේ. තරු ත්‍රිකෝණයේ පිටත ලූපය පේළිය තෝරයි; ඇතුළු ලූපය එම ගණනට තරු මුද්‍රණය කරයි.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  FOR Row = 1 TO 4
    FOR Col = 1 TO Row
      DISPLAY "*" (same line)
    ENDFOR
    Move to a new line
  ENDFOR
END
```



ගැලීම් සටහන - පුනරාවර්තනය තුළ පුනරාවර්තනය: තරු ත්‍රිකෝණය

පුනරාවර්තනය තුළ පුනරාවර්තනය: තරු ත්‍රිකෝණය

Pascal

```
program Stars;
var
  row, col: integer;
begin
  for row := 1 to 4 do
  begin
    for col := 1 to row do
      write('*');
    writeln;
  end;
end.
```

Python

```
for row in range(1, 5):
    for col in range(1, row + 1):
        print("*", end="")
    print()
```

නියැදි ධාවනය

```
*
**
***
****
```

Row	ඇතුළු ලූපය ධාවනය (වාර)	මුද්‍රණය වූ දේ
1	1	*
2	2	**
3	3	***
4	4	****

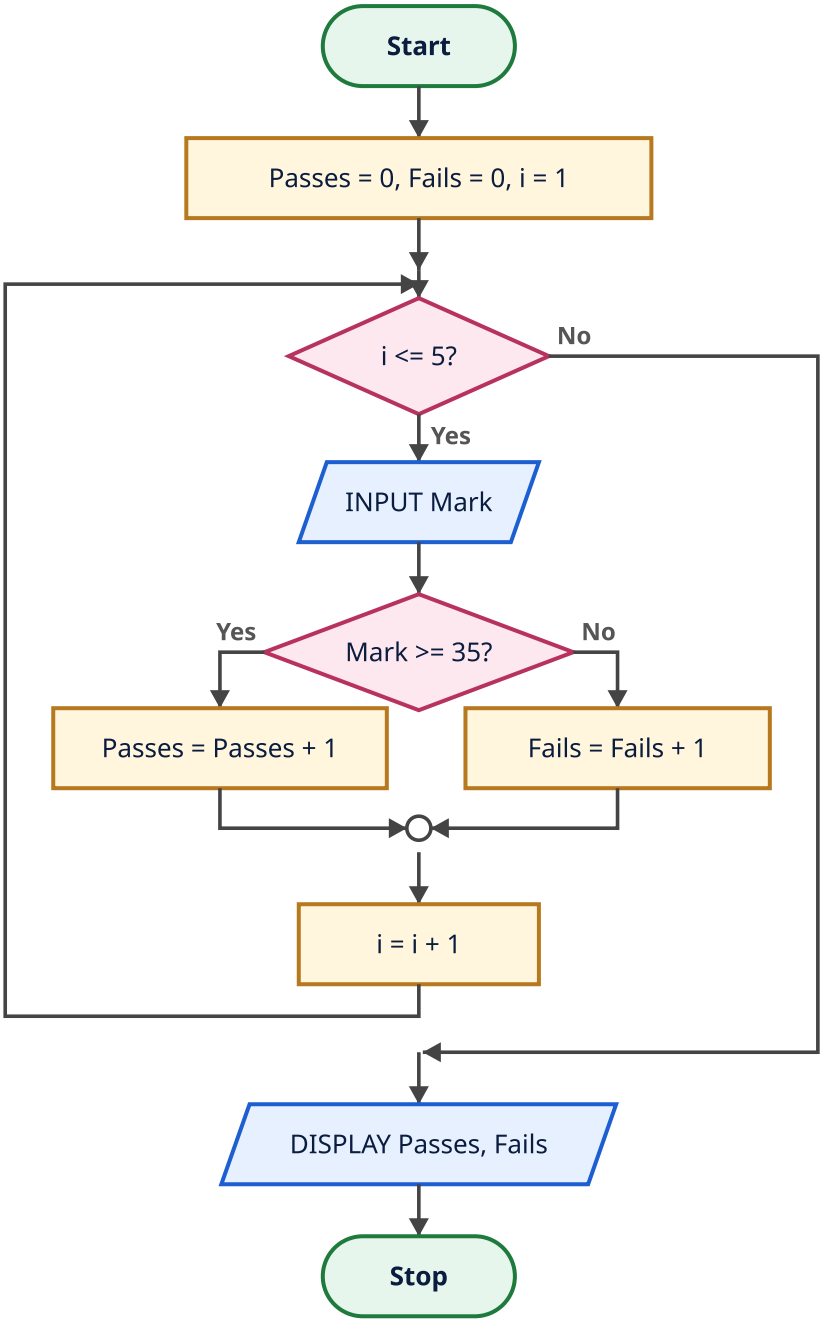
💡 මුළු තරු = 1 + 2 + 3 + 4 = 10. පිටත ලූපය 4 වරක් සහ ඇතුළු ලූපය සැමවිටම 3 වරක් ධාවනය වූයේ නම්, ඇතුළත 4 × 3 = 12 වරක් ධාවනය වේ.

3. ලූපයක් තුළ වරණයක්

ලූපයක් ලකුණු 5ක් කියවයි. ලූපය **තුළ**, IF එකක් එක් එක් ලකුණ සමත්ද (≥ 35) අසමත්ද යන්න තීරණය කර ඒවා ගණන් කරයි.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  Passes = 0
  Fails = 0
  FOR i = 1 TO 5
    INPUT Mark
    IF Mark >= 35 THEN
      Passes = Passes + 1
    ELSE
      Fails = Fails + 1
    ENDIF
  ENDFOR
  DISPLAY Passes, Fails
END
```



ගැලීම් සටහන - පුනරාවර්තනය තුළ වරණය: සමත් සහ අසමත් ගණන් කිරීම

පුනරාවර්තනය තුළ වරණය: සමත් සහ අසමත් ගණන් කිරීම

Pascal

```
program PassCount;
var
  i, mark, passes, fails: integer;
begin
  passes := 0;
  fails := 0;
  for i := 1 to 5 do
  begin
    write('Enter mark ', i, ': ');
    readln(mark);
    if mark >= 35 then
      passes := passes + 1
    else
      fails := fails + 1;
  end;
  writeln('Pass = ', passes);
  writeln('Fail = ', fails);
end.
```

Python

```
passes = 0
fails = 0
for i in range(1, 6):
    mark = int(input(f"Enter mark {i}: "))
    if mark >= 35:
        passes = passes + 1
    else:
        fails = fails + 1
print("Pass =", passes)
print("Fail =", fails)
```

නියැදි ධාවනය

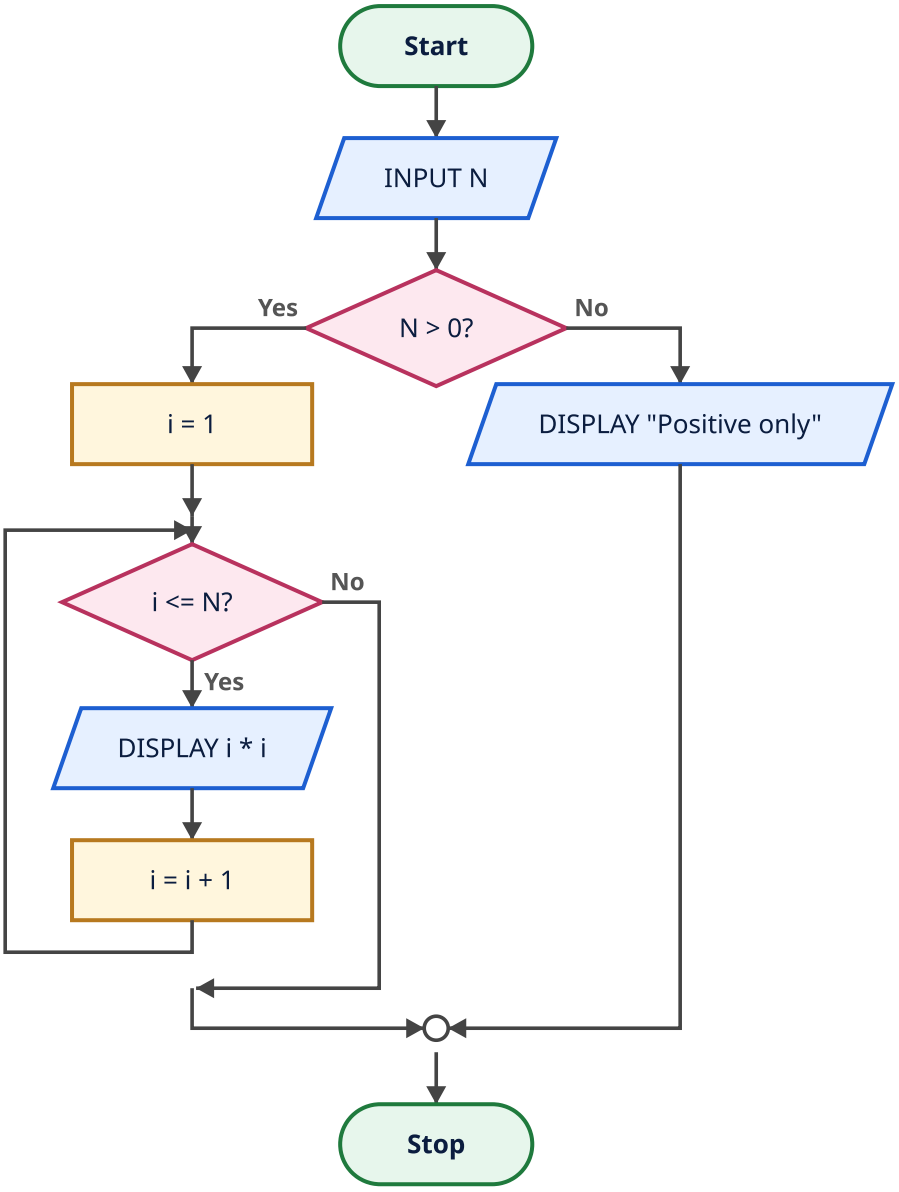
```
Enter mark 1: 45
Enter mark 2: 30
Enter mark 3: 78
Enter mark 4: 22
Enter mark 5: 60
Pass = 3
Fail = 2
```

4. වරණයක් තුළ ලූපයක්

මුලින් ආදානය ධනදැයි පරීක්ෂා කරන්න. **එසේ නම් පමණක්**, 1^2, 2^2, ... වර්ග පෙන්වන ලූපයක් ධාවනය කරන්න.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  INPUT N
  IF N > 0 THEN
    FOR i = 1 TO N
      DISPLAY i * i
    ENDFOR
  ELSE
    DISPLAY "Please enter a positive number"
  ENDIF
END
```



ගැලීම් සටහන – වරණය තුළ පුනරාවර්තනය: ධන සංඛ්‍යාවකට පමණක් වර්ග

වරණය තුළ පුනරාවර්තනය: ධන සංඛ්‍යාවකට පමණක් වර්ග

Pascal

```
program Squares;
var
  n, i: integer;
begin
  write('How many squares? ');
  readln(n);
  if n > 0 then
  begin
    for i := 1 to n do
      writeln('Square of ', i, ' = ', i * i);
  end
  else
    writeln('Please enter a positive number');
end.
```

Python

```
n = int(input("How many squares? "))
if n > 0:
    for i in range(1, n + 1):
        print("Square of", i, "=", i * i)
else:
    print("Please enter a positive number")
```

නියැදි ධාවනය 1

```
How many squares? 4
Square of 1 = 1
Square of 2 = 4
Square of 3 = 9
Square of 4 = 16
```

නියැදි ධාවනය 2

```
How many squares? -2
Please enter a positive number
```

5. කැදැලි කේත සඳහා උපදෙස්

- සෑම ඇතුළු මට්ටමක්ම තව ටිකක් **ඉන්ඩෙන්ට්** කරන්න, එවිට කුමන ප්‍රකාශය කොතැනට අයත්දැයි පෙනේ.
- පිටත එකට පෙර **ඇතුළු** ව්‍යුහය වසන්න (ENDFOR ට පෙර ENDIF).
- Pascal හි කොටසක ප්‍රකාශ එකකට වඩා ඇති විට **begin … end** භාවිත කරන්න.
- කැදැලි ලූප සඳහා වෙනස් ගණක නම් භාවිත කරන්න (Row සහ Col, i සහ j).

ප්‍රධාන පද

කැදැලි කිරීම – එක් පාලන ව්‍යුහයක් තවත් එකක් තුළ තැබීම.

කැදැලි IF – වෙනත් IF එකක් තුළ ඇති IF එකක්.

කැදැලි ලූපය – වෙනත් ලූපයක් තුළ ඇති ලූපයක්.

පිටත ලූපය – වෙනත් ලූපයක් අඩංගු ලූපය.

ඇතුළු ලූපය – ඇතුළත ලූපය; පිටතේ සෑම ධාවනයකටම සම්පූර්ණයෙන් ධාවනය වේ.

ඉන්ඩෙන්ටේෂන් – ප්‍රකාශයක් අයත් මට්ටම පෙන්වන හිස්තැන්.

begin … end – ප්‍රකාශ කිහිපයක් එක් කාණ්ඩයකට එක් කරන Pascal වචන.

විශාලතම – කණ්ඩායමක ඇති විශාලතම අගය.

සමත් ගණන – සෑම සමත් වීමකටම 1 ක් එකතු කරන ගණකය.

වර්ගය – සංඛ්‍යාවක් එයින්ම ගුණ කිරීම, උදා 4 × 4 = 16.

ධන සංඛ්‍යාව – 0 ට වැඩි සංඛ්‍යාවක්.

මට්ටම – ව්‍යුහයක් අන් ඒවා තුළ කෙතරම් ගැඹුරින් ඇත්ද යන්න.

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. කැදැලි ව්‍යුහයක් යනු කුමක්ද?

පිළිතුර: වෙනත් පාලන ව්‍යුහයක් තුළ තැබූ පාලන ව්‍යුහයකි.

2. A = 12, B = 45, C = 30 සඳහා විශාලතම-තුනෙන් වැඩසටහන පෙන්වන්නේ කුමක්ද?

පිළිතුර: Largest = 45.

3. පිටත ලූපය 3 වරක් සහ ඇතුළු ලූපය සෑම විටම 4 වරක් ධාවනය වේ නම් ඇතුළත කී වරක් ධාවනය වේද?

පිළිතුර: 12 වරක්.

4. ලකුණු 45, 30, 78, 22, 60 – සමත් සහ අසමත් කීයද?

පිළිතුර: සමත් = 3, අසමත් = 2.

5. squares වැඩසටහන −2 සඳහා පෙන්වන්නේ කුමක්ද?

පිළිතුර: Please enter a positive number.

6. කැදැලි කේත ඉන්ඩෙන්ට් කරන්නේ ඇයි?

පිළිතුර: කුමන ප්‍රකාශ කුමන ව්‍යුහයට අයත්දැයි පැහැදිලිව දැකීමට.

බහුවරණ ප්‍රශ්න

1. වෙනත් ලූපයක් තුළ තැබූ ලූපයක්...

A. නියතයකි
B. sentinel එකකි
C. කැදැලි ලූපයකි
D. case එකකි

2. A = 50, B = 20, C = 35 සඳහා විශාලතම...

A. 50
B. 35
C. 105
D. 20

3. විශාලතම-තුනෙන් ගැලීම් සටහනේ පළමු ප්‍රශ්නය...

A. A > C?
B. C > 0?
C. B > C?
D. A > B?

4. පිටත ලූපය 4 වරක්, ඇතුළු ලූපය සෑම විටම 3 වරක්. ඇතුළත ධාවනය වන්නේ...

A. 3 වරක්
B. 12 වරක්
C. 4 වරක්
D. 7 වරක්

5. තරු ත්‍රිකෝණයේ අවසාන පේළිය (Row = 4)...

A. *
B. ****
C. **
D. ***

6. පේළි 4ක ත්‍රිකෝණයේ මුළු තරු...

A. 10
B. 4
C. 16
D. 8

7. තරු වැඩසටහනේ ඇතුළු ලූපය ධාවනය වන්නේ...

A. Col = 0 තාක්
B. Row > Col + 5 තාක්
C. Col <= Row තාක්
D. Row <= 4 පමණක් තාක්

8. ලකුණු 45, 30, 78, 22, 60 අනුව Pass = ...

A. 2
B. 4
C. 5
D. 3

9. ලූපයක් තුළ සමත් ගණන් කිරීම ... සඳහා උදාහරණයකි

A. ලූපයක් තුළ වරණය
B. වරණයක් තුළ ලූපය
C. කැදැලි අරාවක්
D. අනුක්‍රමය පමණි

10. squares වැඩසටහන මුලින් සංඛ්‍යාව පරීක්ෂා කර පසුව ලූප කරයි. මෙය...

A. ලූපයක් තුළ වරණයකි
B. case ප්‍රකාශයකි
C. ලූපයක් තුළ ලූපයකි
D. වරණයක් තුළ ලූපයකි

11. ආදානය 4 සඳහා squares වැඩසටහනේ අවසාන පේළිය...

A. Square of 4 = 8
B. Square of 4 = 16
C. Square of 3 = 9
D. Square of 16 = 4

12. මුලින්ම වැසිය යුත්තේ කුමක්ද?

A. පිටත ව්‍යුහය
B. වැඩසටහන
C. ඇතුළු ව්‍යුහය
D. ඕනෑම එකක්

13. Pascal හි IF එකක් තුළ ප්‍රකාශ කිහිපයක් කාණ්ඩ කළ යුත්තේ...

A. [ ]
B. begin ... end
C. { }
D. ( )

14. කැදැලි ලූප ගණක දෙකකට හොඳ නම්...

A. දෙකම Total
B. x and x
C. Row and Col
D. දෙකම i

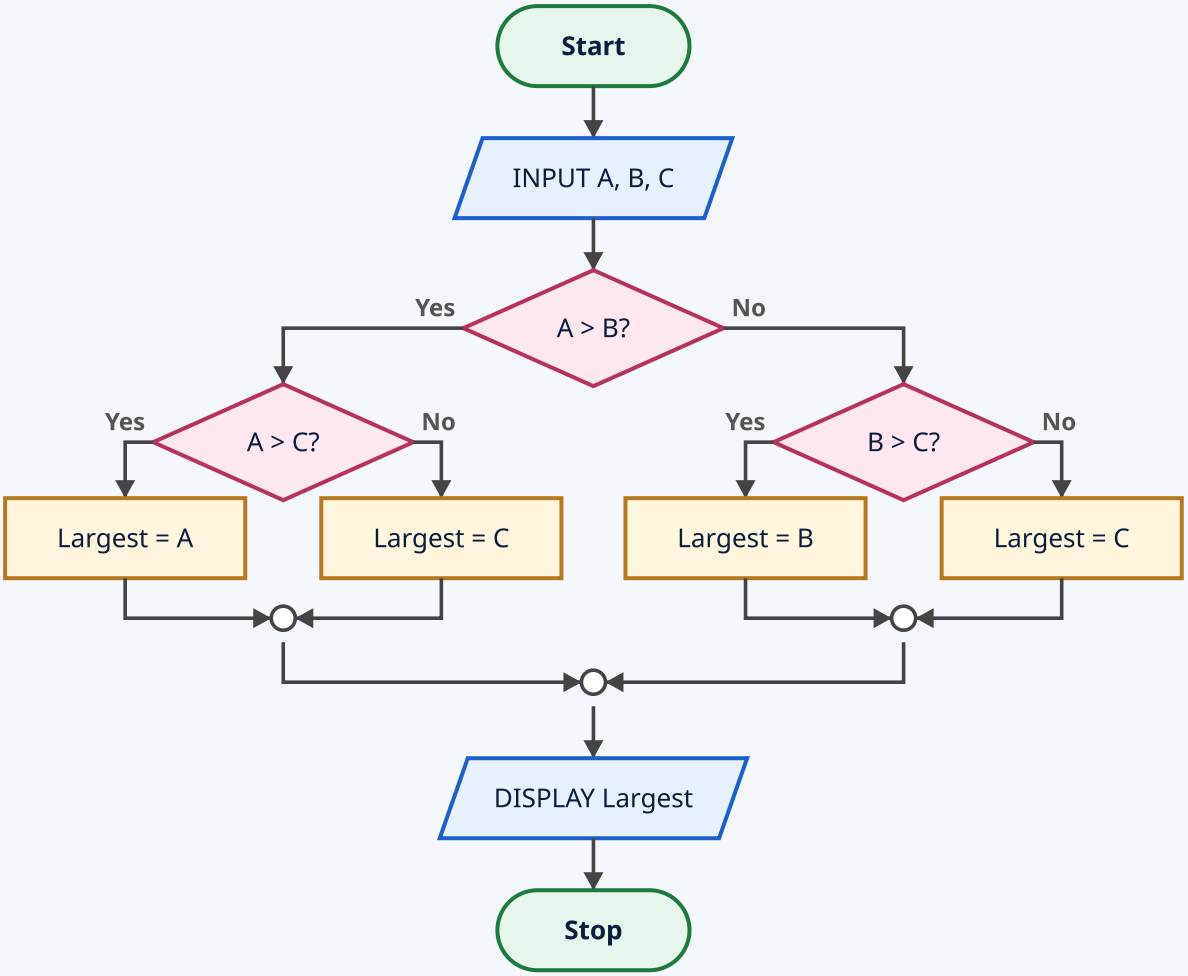
15. කැදැලි කිරීම යනු...

A. එක් ව්‍යුහයක් තවත් එකක් තුළ තැබීම
B. අනුක්‍රමය පමණක් භාවිතය
C. ලූපයක් මකා දැමීම
D. වැඩසටහන් දෙකක් එකතු කිරීම

ව්‍යුහගත ප්‍රශ්න

1. කැදැලි IF භාවිතයෙන් A, B සහ C සංඛ්‍යා තුනක් ආදානය කර විශාලතම එක පෙන්වන ගැලීම් සටහනක් අඳින්න. (ලකුණු 5) *(ලකුණු 5)*

Start/ආදානය (1); A > B? (1); ඔව් → A > C? (1); නැත → B > C? (1); Largest පැවරීම සහ පෙන්වීම, Stop (1).



ගැලීම් සටහන – වරණය තුළ වරණය: තුනෙන් විශාලතම

2. *, , *, ** ත්‍රිකෝණය මුද්‍රණය කිරීමට කැදැලි ලූප භාවිත කරන වැඩසටහනක් (Pascal හෝ Python) ලියන්න. (ලකුණු 4)** *(ලකුණු 4)*

පිටත ලූපය Row 1 සිට 4 (1); ඇතුළු ලූපය Col 1 සිට Row (1); නව පේළියක් නැතිව * මුද්‍රණය (1); ඇතුළු ලූපයට පසු නව පේළිය (1).

පුනරාවර්තනය තුළ පුනරාවර්තනය: තරු ත්‍රිකෝණය

Pascal

```
program Stars;
var
  row, col: integer;
begin
  for row := 1 to 4 do
  begin
    for col := 1 to row do
      write('*');
    writeln;
  end;
end.
```

Python

```
for row in range(1, 5):
    for col in range(1, row + 1):
        print("*", end="")
    print()
```

නියැදි ධාවනය

```
*
**
***
****
```

3. ලකුණු 5ක් කියවා සමත් (≥ 35) කීයද අසමත් කීයද යන්න ගණන් කරන ව්‍යාජ කේතය ලියන්න. (ලකුණු 5) *(ලකුණු 5)*

PassCount = 0, FailCount = 0 (1); FOR k = 1 TO 5 (1); INPUT Mark (1); IF Mark >= 35 PassCount ට 1 ELSE FailCount ට 1 (1); දෙකම පෙන්වීම (1).

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  Passes = 0
  Fails = 0
  FOR i = 1 TO 5
    INPUT Mark
    IF Mark >= 35 THEN
      Passes = Passes + 1
    ELSE
      Fails = Fails + 1
    ENDIF
  ENDFOR
  DISPLAY Passes, Fails
END
```

4. කැදැලි කිරීමේ වර්ග හතර නම් කර එක් එකකට උදාහරණයක් දෙන්න. (ලකුණු 4) *(ලකුණු 4)*

වරණය තුළ වරණය – තුනෙන් විශාලතම; ලූපය තුළ ලූපය – තරු ත්‍රිකෝණය; ලූපය තුළ වරණය – සමත් ගණන් කිරීම; වරණය තුළ ලූපය – ධන සංඛ්‍යාවකට පමණක් වර්ග (එකට 1).

5. තරු වැඩසටහන (පේළි 4) සඳහා Row, ඇතුළු ලූපය ධාවනය වන වාර ගණන සහ මුද්‍රණය වන දේ පෙන්වන අනුරේඛන වගුවක් සම්පූර්ණ කරන්න. (ලකුණු 3) *(ලකුණු 3)*

Row 1 → 1 → *; Row 2 → 2 → **; Row 3 → 3 → *; Row 4 → 4 → **** (සියල්ල නිවැරදි නම් 3).

Row	ඇතුළු ලූපය ධාවනය (වාර)	මුද්‍රණය වූ දේ
1	1	*
2	2	**
3	3	***
4	4	****

සාරාංශය

- කැදැලි = ව්‍යුහයක් තුළ තවත් එකක්.
- කැදැලි IF: තුනෙන් විශාලතම.
- කැදැලි ලූපය: ඇතුළු ලූපය සෑම විටම සම්පූර්ණයෙන් ධාවනය වේ.
- ඇතුළත ධාවන = පිටත × ඇතුළු (ස්ථාවර ප්‍රමාණ).

- ලූපයක් තුළ IF: සමත් ගණන් කිරීම.
- IF එකක් තුළ ලූපය: වලංගු නම් පමණක් ධාවනය.
- සෑම මට්ටමක්ම ඉන්ඩෙන්ට් කරන්න; ඇතුළත මුලින් වසන්න.
- Pascal: begin … end ප්‍රකාශ කාණ්ඩ කරයි.

MCQ පිළිතුරු – 1.8

1. **C** – කැදැලි.
2. **A** – A විශාලතමයි.
3. **D** – පිටත IF.
4. **B** – 4 × 3.
5. **B** – තරු 4.
6. **A** – 1+2+3+4.
7. **C** – ඇතුළු කොන්දේසිය.
8. **D** – 45, 78, 60.
9. **A** – FOR තුළ IF.
10. **D** – IF තුළ FOR.
11. **B** – 4 × 4.
12. **C** – ඇතුළත මුලින්.
13. **B** – begin … end.
14. **C** – වෙනස් නම්.
15. **A** – අර්ථය.

© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.

G.C.E. O/L ICT · 11 ශ්‍රේණිය · පාඩම 1

1.9 ඒක-මාන අරා

විෂය නිර්දේශය 10.9. සිසුන් 40 දෙනෙකුගේ ලකුණු වෙනම විචල්‍ය 40 ක – Mark1, Mark2 … Mark40 – ගබඩා කිරීම ගැන සිතන්න. ඉතා අවුල්! **අරාවක්** (array) **එකම දත්ත වර්ගයේ** අගයන් බොහොමයක් **එක් නමක්** යටතේ ගබඩා කරයි. සෑම අගයක්ම **අවයවයකි** (element), අප එයට එහි **දර්ශකය** (index – ස්ථාන අංකය) මගින් ළඟා වෙමු, Marks[3] වැනි. අරා **එක ළඟ පිහිටි** (contiguous) මතකයේ තබන නිසා ඕනෑම අවයවයකට **කෙළින්ම** ළඟා විය හැක (random access).

1. අරා අවශ්‍ය ඇයි?

- බොහෝ අගයන්ට **එක් නමක්** – කියවීමට පහසුයි.
- පේළි කිහිපයකින් සියලු අවයව සැකසීමට **ලූපයක්** භාවිත කරන්න.
- එකතුව, සාමාන්‍යය, ඉහළම අගය සොයා ගැනීමට, සෙවීමට හෝ පිළිවෙළට සැකසීමට පහසුයි.
- සිසුන් 5 සිට 40 දක්වා වෙනස් කිරීමට එක් අංකයක් පමණක් වෙනස් කරන්න.

2. දර්ශකය, අවයව සහ මතකය

පාසල් කොරිඩෝවක අංක කළ ලොකර් පේළියක් ගැන සිතන්න. සියලු ලොකර් **එක ළඟ** (contiguous) ඇති අතර 1, 2, 3 මුලින් නොවිවෘත කර ලොකර් 4 **කෙළින්ම** විවෘත කළ හැක (random access). අවයව 5ක් ඇති Marks අරාව:

Pascal දර්ශකය	1	2	3	4	5
Python දර්ශකය	0	1	2	3	4
අගය	65	80	45	90	70

💡 **දර්ශකය ආරම්භය:** Pascal හි අප එය තෝරමු (array[1..5] 1 න් ආරම්භ වේ). **Python** හි list එකක් සැමවිටම **0** න් ආරම්භ වේ, එබැවින් පළමු ලකුණ marks[0] ද පස්වැන්න marks[4] ද වේ. O/L ව්‍යාජ කේතය සාමාන්‍යයෙන් 1 න් ආරම්භ වේ.

3. ප්‍රකාශ කිරීම, පැවරීම සහ ප්‍රවේශය

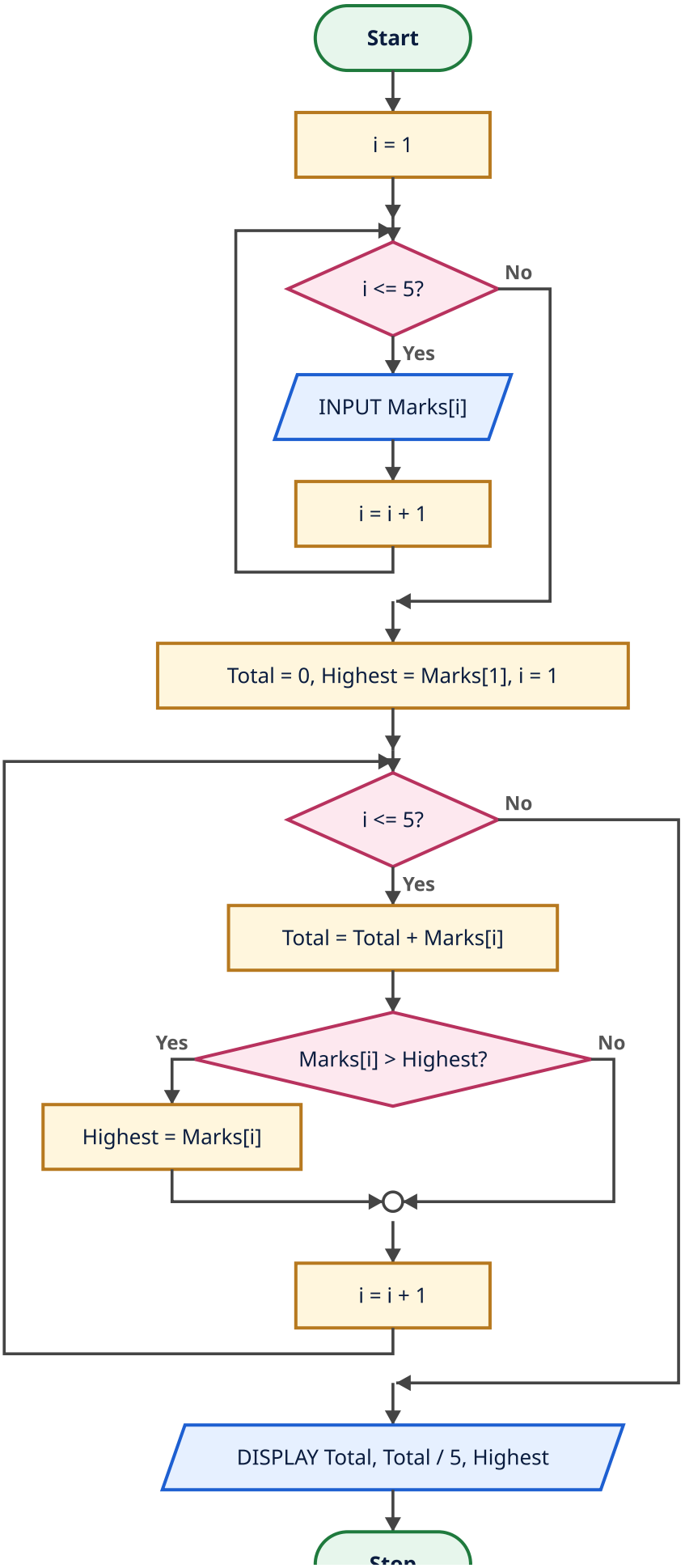
ක්‍රියාව	Pascal	Python
ප්‍රකාශ කිරීම (පූර්ණ සංඛ්‍යා 5)	var Marks: array[1..5] of integer;	marks = [0, 0, 0, 0, 0]
අගයක් පැවරීම	Marks[3] := 45;	marks[2] = 45
අගයක් කියවීම	writeln(Marks[3]);	print(marks[2])

ලූපයකින් පිරවීම	for i := 1 to 5 do readln(Marks[i]);	for i in range(5): marks[i] = int(input())

4. උදාහරණය: එකතුව, සාමාන්‍යය සහ ඉහළම

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  FOR i = 1 TO 5
    INPUT Marks[i]
  ENDFOR
  Total = 0
  Highest = Marks[1]
  FOR i = 1 TO 5
    Total = Total + Marks[i]
    IF Marks[i] > Highest THEN
      Highest = Marks[i]
    ENDIF
  ENDFOR
  DISPLAY Total, Total / 5, Highest
END
```


Start
i = 1
i <= 5?
No
Yes
INPUT Marks[i]
i = i + 1
Total = 0, Highest = Marks[1], i = 1
i <= 5?
No
Yes
Total = Total + Marks[i]
Yes
Marks[i] > Highest?
No
Highest = Marks[i]
i = i + 1
DISPLAY Total, Total / 5, Highest

Stop

ගැලීම් සටහන – අරාව: ලකුණු 5ක් ගබඩා කර එකතුව, සාමාන්‍යය, ඉහළම

අරාව: ලකුණු 5ක් ගබඩා කර එකතුව, සාමාන්‍යය, ඉහළම

Pascal

```
program MarksArray;
var
  marks: array[1..5] of integer;
  i, total, highest: integer;
begin
  for i := 1 to 5 do
  begin
    write('Enter mark ', i, ': ');
    readln(marks[i]);
  end;
  total := 0;
  highest := marks[1];
  for i := 1 to 5 do
  begin
    total := total + marks[i];
    if marks[i] > highest then
      highest := marks[i];
  end;
  writeln('Total = ', total);
  writeln('Average = ', total / 5:0:2);
  writeln('Highest = ', highest);
end.
```

Python

```
marks = [0, 0, 0, 0, 0]          # 5 elements, index 0 to 4
for i in range(5):
    marks[i] = int(input(f"Enter mark {i + 1}: "))
total = 0
highest = marks[0]
for i in range(5):
    total = total + marks[i]
    if marks[i] > highest:
        highest = marks[i]
print("Total =", total)
print(f"Average = {total / 5:.2f}")
print("Highest =", highest)
```

නියැදි ධාවනය

```
Enter mark 1: 65
Enter mark 2: 80
Enter mark 3: 45
Enter mark 4: 90
Enter mark 5: 70
Total = 350
Average = 70.00
Highest = 90
```

i	Marks[i]	Total	Marks[i] > Highest ?	Highest

1	65	65	නැත	65
2	80	145	ඔව්	80
3	45	190	නැත	80
4	90	280	ඔව්	90
5	70	350	නැත	90

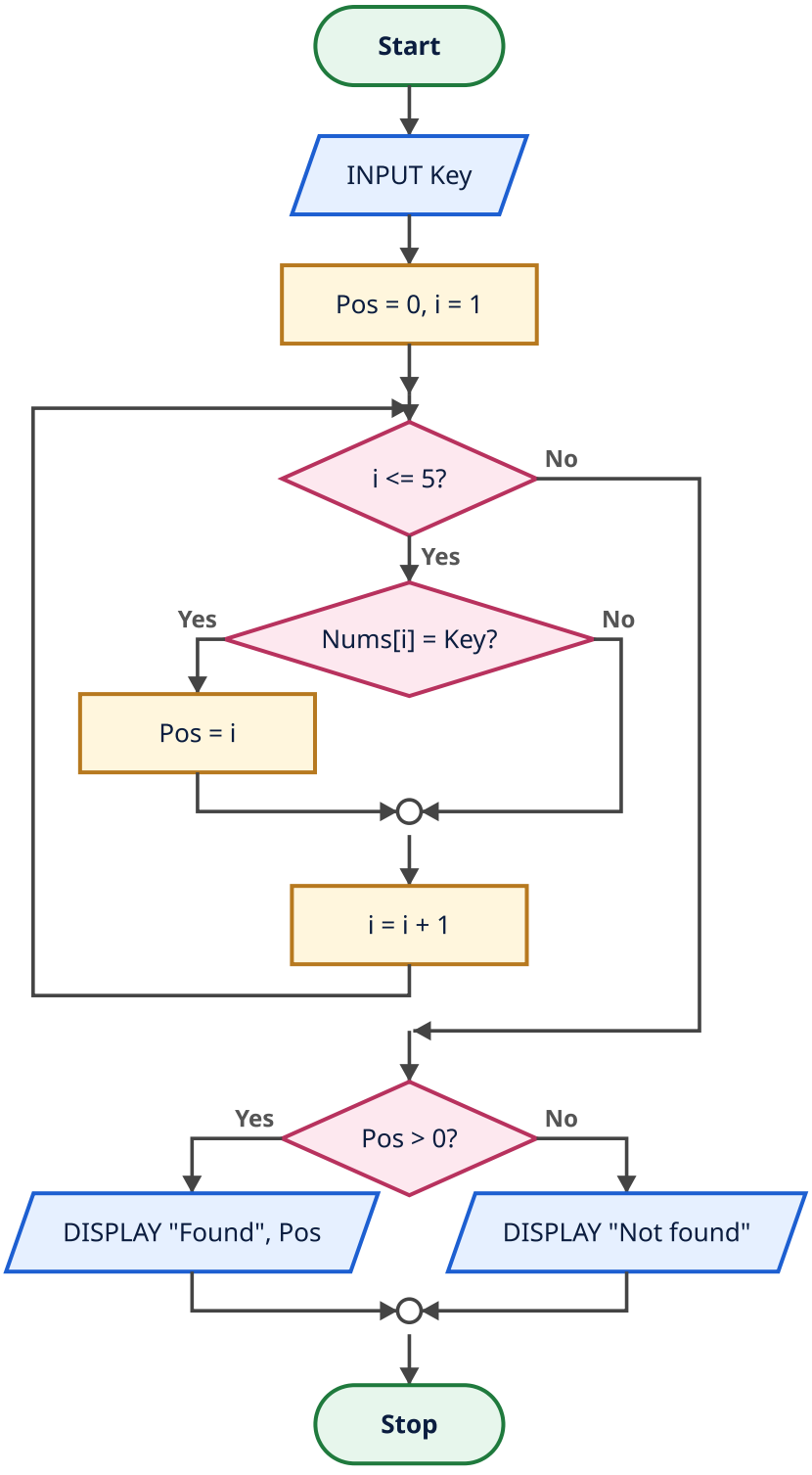
Total = 350 සහ Highest = 90, සැබෑ වැඩසටහන් ප්‍රතිදානයට සමානයි. සාමාන්‍යය = 350 ÷ 5 = 70.00.

5. උදාහරණය: අරාවක සෙවීම (රේඛීය සෙවීම)

Nums අරාවේ 12, 45, 7, 30, 18 ඇත. සංඛ්‍යාවක් සොයා ගැනීමට, සෑම අවයවයක්ම **පළමුවැන්නේ සිට අවසානය දක්වා** පරීක්ෂා කරන්න. මෙය **රේඛීය සෙවීම** (linear search) ලෙස හැඳින්වේ.

ව්‍යාජ කේතය (Pseudocode)

```
BEGIN
  Nums = [12, 45, 7, 30, 18]
  INPUT Key
  Pos = 0
  FOR i = 1 TO 5
    IF Nums[i] = Key THEN
      Pos = i
    ENDIF
  ENDFOR
  IF Pos > 0 THEN
    DISPLAY "Found at position ", Pos
  ELSE
    DISPLAY "Not found"
  ENDIF
END
```



ගැලීම් සටහන – අරාව: සංඛ්‍යාවක් සෙවීම

අරාව: සංඛ්‍යාවක් සෙවීම

Pascal

```
program SearchArray;
var
  nums: array[1..5] of integer;
  i, key, pos: integer;
begin
  nums[1] := 12;
  nums[2] := 45;
  nums[3] := 7;
  nums[4] := 30;
  nums[5] := 18;
  write('Enter number to find: ');
  readln(key);
  pos := 0;
  for i := 1 to 5 do
    if nums[i] = key then
      pos := i;
  if pos > 0 then
    writeln('Found at position ', pos)
  else
    writeln('Not found');
end.
```

Python

```
nums = [12, 45, 7, 30, 18]
key = int(input("Enter number to find: "))
pos = 0
for i in range(5):
    if nums[i] == key:
        pos = i + 1      # position counted from 1
if pos > 0:
    print("Found at position", pos)
else:
    print("Not found")
```

නියැදි ධාවනය 1

```
Enter number to find: 30
Found at position 4
```

නියැදි ධාවනය 2

```
Enter number to find: 50
Not found
```

i	Nums[i]	Nums[i] = 30 ?	Pos
1	12	නැත	0
2	45	නැත	0
3	7	නැත	0
4	30	ඔව්	4
5	18	නැත	4

6. ස්ථානයක් සොයා ගැනීම

සතියක කොළඹ උෂ්ණත්ව (°C) අරාවක ගබඩා කර ඇත. වැඩසටහන ඉහළම අගය පමණක් නොව එහි **දර්ශකයද** තබා ගන්නා නිසා **කුමන දිනය** උණුසුම්ම වූයේදැයි කිව හැක.

ඉහළම උෂ්ණත්වයේ ස්ථානය

Pascal

```
program HottestDay;
var
  t: array[1..7] of integer;
  i, best: integer;
begin
  t[1] := 31; t[2] := 33; t[3] := 30; t[4] := 34;
  t[5] := 32; t[6] := 29; t[7] := 33;
  best := 1;
  for i := 2 to 7 do
    if t[i] > t[best] then
      best := i;
  writeln('Day ', best, ': ', t[best]);
end.
```

Python

```
t = [31, 33, 30, 34, 32, 29, 33]
best = 0
for i in range(1, 7):
    if t[i] > t[best]:
        best = i
print("Day " + str(best + 1) + ":", t[best])
```

නියැදි ධාවනය

```
Day 4: 34
```

ප්‍රධාන පද

අරාව – එක් නමක් යටතේ එකම වර්ගයේ අගයන් එකතුවක්.

ඒක-මාන අරාව – අවයවවල තනි පේළියක් (ලැයිස්තුවක්) වන අරාව.

අවයවය – අරාවක ගබඩා කළ එක් අගයක්.

දර්ශකය – අවයවයක ස්ථාන අංකය, උදා Marks[3] හි 3.

එක ළඟ පිහිටි මතකය – මතකයේ එක ළඟ ගබඩා කළ අවයව.

සෘජු ප්‍රවේශය – දර්ශකය භාවිතයෙන් ඕනෑම අවයවයකට වහාම ළඟා වීම.

ප්‍රකාශ කිරීම – නම, ප්‍රමාණය සහ වර්ගය සමඟ අරාව සෑදීම.

පැවරීම – අවයවයක අගයක් ගබඩා කිරීම, Marks[3] := 45.

ප්‍රවේශය – අවයවයක අගය කියවීම.

List (Python) – Python හි අරා වැනි වර්ගය; දර්ශකය 0 න් ආරම්භ වේ.

රේඛීය සෙවීම – ආරම්භයේ සිට එකින් එක අවයව පරීක්ෂා කිරීම.

අරා ප්‍රමාණය – අරාවක ඇති අවයව ගණන.

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. අරාවක් යනු කුමක්ද?

පිළිතුර: එක් නමක් යටතේ ගබඩා කළ එකම දත්ත වර්ගයේ අගයන් කණ්ඩායමකි.

2. දර්ශකයක් යනු කුමක්ද?

පිළිතුර: අවයවයකට ළඟා වීමට භාවිත කරන ස්ථාන අංකයයි.

3. අරා සඳහා “random access” යන්නෙහි තේරුම කුමක්ද?

පිළිතුර: දර්ශකය මගින් ඕනෑම අවයවයකට කෙළින්ම ළඟා විය හැක.

4. Pascal හි අවයව 5ක පූර්ණ සංඛ්‍යා අරාවක් ප්‍රකාශ කරන්න.

පිළිතුර: var Marks: array[1..5] of integer;

5. Python හි පළමු අවයවයේ දර්ශකය කුමක්ද?

පිළිතුර: 0.

6. ලකුණු 65, 80, 45, 90, 70 සඳහා ඉහළම කුමක්ද?

පිළිතුර: 90.

බහුවරණ ප්‍රශ්න

1. අරාවක් බොහෝ අගයන් ගබඩා කරන්නේ...

A. එකම දත්ත වර්ගයේ
B. පාඨ පමණි
C. Boolean පමණි
D. සෑමවිටම වෙනස් දත්ත වර්ගවල

2. අරාවක් තුළ ඇති එක් අගයක් හඳුන්වන්නේ...

A. කාරකය
B. අවයවය
C. දර්ශකය
D. හඳුන්වනය

3. අවයවයක ස්ථාන අංකය එහි...

A. දර්ශකයයි
B. අගයයි
C. වර්ගයයි
D. ප්‍රමාණයයි

4. අරා අවයව ගබඩා වන්නේ...

A. වෙනම ගොනුවල
B. තැටියේ අහඹු ස්ථානවල
C. එක ළඟ පිහිටි මතකයේ
D. මුද්‍රකයේ

5. Marks[1..3] නොකියවා Marks[4] ට ළඟා වීම හඳුන්වන්නේ...

A. කැදැලි කිරීම
B. අනුක්‍රමික ප්‍රවේශය
C. සෘජු ප්‍රවේශය
D. සම්පාදනය

6. පූර්ණ සංඛ්‍යා 5ක අරාවක නිවැරදි Pascal ප්‍රකාශනය:

A. A = [5]
B. array A of 5;
C. var A: integer[5];
D. var A: array[1..5] of integer;

7. Python හි marks හි පළමු අවයවය...

A. marks(1)
B. marks[0]
C. marks[1]
D. marks{0}

8. අවයව 5ක Python list එකක අවසාන දර්ශකය...

A. 4
B. 1
C. 6
D. 5

9. Marks = 65, 80, 45, 90, 70 (Pascal). Marks[3]...

A. 90
B. 45
C. 3
D. 80

10. 65, 80, 45, 90 සහ 70 හි එකතුව...

A. 90
B. 340
C. 350
D. 70

11. Nums = 12, 45, 7, 30, 18. 30 සෙවීමෙන් ලැබෙන්නේ...

A. Found at position 4
B. Not found
C. Found at position 3
D. Found at position 30

12. එම අරාවේම 50 සෙවීමෙන් ලැබෙන්නේ...

A. Found at position 1
B. දෝෂයක්
C. Not found
D. Found at position 5

13. ආරම්භයේ සිට එකින් එක අවයව පරීක්ෂා කිරීම...

A. අහඹු සෙවීමකි
B. කැදැලි IF එකකි
C. නියතයකි
D. රේඛීය සෙවීමකි

14. Python හි තුන්වන අවයවයට 45 පවරන ප්‍රකාශය කුමක්ද?

A. 45 = marks[2]
B. marks = 45
C. marks[3] = 45
D. marks[2] = 45

15. උෂ්ණත්ව 31, 33, 30, 34, 32, 29, 33. උණුසුම්ම දිනය...

A. Day 6: 29
B. Day 4: 34
C. Day 7: 33
D. Day 2: 33

ව්‍යුහගත ප්‍රශ්න

1. (i) ඒක-මාන අරාවක් යනු කුමක්ද? (ලකුණු 1)
(ii) වෙනම විචල්‍ය වෙනුවට අරාවක් භාවිතයේ වාසි දෙකක් දෙන්න. (ලකුණු 2) *(ලකුණු 3)*

> (i) එක් නමක් යටතේ එකම වර්ගයේ අවයවවල තනි පේළියක්. (ii) බොහෝ අගයන්ට එක් නමක්; ලූපයකින් සියල්ල සැකසිය හැක; ප්‍රමාණය වෙනස් කිරීම පහසුයි (ඕනෑම දෙකක්).

2. (a) දර්ශකය (b) එක ළඟ පිහිටි මතකය (c) සෘජු ප්‍රවේශය යන පද පහදන්න. (ලකුණු 3) *(ලකුණු 3)*

(a) අවයවයක ස්ථාන අංකය. (b) අවයව මතකයේ එක ළඟ ගබඩා වේ. (c) දර්ශකය භාවිතයෙන් ඕනෑම අවයවයකට කෙළින්ම ළඟා විය හැක.

3. ලකුණු 5ක් අරාවකට කියවා එකතුව, සාමාන්‍යය සහ ඉහළම ලකුණ පෙන්වන වැඩසටහනක් (Pascal හෝ Python) ලියන්න. (ලකුණු 6) *(ලකුණු 6)*

අරාව ප්‍රකාශ කිරීම (1); කියවීමට ලූපය (1); total = 0, highest = පළමු අවයවය (1); එකතු කිරීම සහ සැසඳීමේ ලූපය (2); total, average, highest පෙන්වීම (1).

අරාව: ලකුණු 5ක් ගබඩා කර එකතුව, සාමාන්‍යය, ඉහළම

Pascal

```
program MarksArray;
var
  marks: array[1..5] of integer;
  i, total, highest: integer;
begin
  for i := 1 to 5 do
  begin
    write('Enter mark ', i, ': ');
    readln(marks[i]);
  end;
  total := 0;
  highest := marks[1];
  for i := 1 to 5 do
  begin
    total := total + marks[i];
    if marks[i] > highest then
      highest := marks[i];
  end;
  writeln('Total = ', total);
  writeln('Average = ', total / 5:0:2);
  writeln('Highest = ', highest);
end.
```

Python

```
marks = [0, 0, 0, 0, 0]          # 5 elements, index 0 to 4
for i in range(5):
    marks[i] = int(input(f"Enter mark {i + 1}: "))
total = 0
highest = marks[0]
for i in range(5):
    total = total + marks[i]
    if marks[i] > highest:
        highest = marks[i]
print("Total =", total)
print(f"Average = {total / 5:.2f}")
print("Highest =", highest)
```

නියැදි ධාවනය

```
Enter mark 1: 65
Enter mark 2: 80
Enter mark 3: 45
Enter mark 4: 90
Enter mark 5: 70
Total = 350
Average = 70.00
Highest = 90
```

4. Nums = 12, 45, 7, 30, 18. 30 සඳහා රේඛීය සෙවීමක අනුරේඛන වගුවක් සම්පූර්ණ කර ප්‍රතිදානය දෙන්න. (ලකුණු 4) *(ලකුණු 4)*

i = 1…5 සමඟ Nums[i] = 30? නැත, නැත, නැත, ඔව්, නැත; Pos = 0, 0, 0, 4, 4 (3); ප්‍රතිදානය Found at position 4 (1).

i	Nums[i]	Nums[i] = 30 ?	Pos
1	12	නැත	0
2	45	නැත	0
3	7	නැත	0
4	30	ඔව්	4
5	18	නැත	4

5. (a) Marks හි තුන්වන අවයවයේ 45 ගබඩා කිරීමට (b) පළමු අවයවය පෙන්වීමට Pascal සහ Python ප්‍රකාශ ලියන්න. (ලකුණු 4) *(ලකුණු 4)*

(a) Pascal Marks[3] := 45; Python marks[2] = 45. (b) Pascal writeln(Marks[1]); Python print(marks[0]) (එකට 1).

සාරාංශය

- අරාව: බොහෝ අගයන්, එකම වර්ගය, එක් නමක්.
- අවයවය = එක් අගයක්; දර්ශකය = එහි ස්ථානය.
- එක ළඟ මතකය → සෘජු ප්‍රවේශය.
- Pascal: array[1..5] of integer.
- Python list දර්ශකය 0 න් ආරම්භ වේ.
- අරාවක් පිරවීමට සහ සැකසීමට ලූපයක් භාවිත කරන්න.
- එක් ලූපයකින් එකතුව, සාමාන්‍යය, ඉහළම.
- රේඛීය සෙවීම අවයව එකින් එක පරීක්ෂා කරයි.

MCQ පිළිතුරු – 1.9

1. **A** – එකම වර්ගය.
2. **B** – අවයවය.
3. **A** – දර්ශකය.
4. **C** – එක ළඟ.
5. **C** – සෘජු.
6. **D** – Pascal වාක්‍ය රීතිය.
7. **B** – 0 සිට දර්ශකය.
8. **A** – 0 සිට 4.
9. **B** – තුන්වන අවයවය.
10. **C** – එකතුව.

11. **A** – හතරවන අවයවය.
12. **C** – අරාවේ නැත.
13. **D** – රේඛීය.
14. **D** – දර්ශකය 2 = තුන්වන.
15. **B** – 34 හතරවැන්නයි.

© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.

G.C.E. O/L ICT · 11 ශ්‍රේණිය · පාඩම 1

1.10 උප වැඩසටහන්: ශ්‍රිත සහ ක්‍රියා පටිපාටි

විෂය නිර්දේශය 10.10. උප වැඩසටහනක් යනු **එක් කාර්යයක්** කරන, නමක් ඇති වැඩසටහනක කුඩා කොටසකි. අප එය එක් වරක් ලියා අවශ්‍ය සෑම විටම **කැඳවමු** (භාවිත කරමු) – නැවත නොලියා දිනපතා අනුගමනය කළ හැකි පොල් සම්බෝල වට්ටෝරුවක් මෙන්. විශාල වැඩසටහන් කුඩා උප වැඩසටහන් බොහොමයකින් සාදා ඇත.

1. උප වැඩසටහන් භාවිත කරන්නේ ඇයි?

නැවත භාවිතය
එක් වරක් ලියා බොහෝ වර කැඳවන්න. එකම කේතය නැවත පිටපත් කිරීම අවශ්‍ය නැත.

කියවීමේ පහසුව
PrintLine, GetGrade වැනි නම් ප්‍රධාන වැඩසටහන තේරුම් ගැනීමට පහසු කරයි.

නඩත්තුවේ පහසුව
කාර්යයක් නිවැරදි කිරීමට හෝ වෙනස් කිරීමට එක් තැනක පමණක් වෙනස් කරන්න.

කණ්ඩායම් වැඩ සහ පරීක්ෂාව
විවිධ අයට විවිධ උප වැඩසටහන් ලියා පරීක්ෂා කළ හැක.

2. උප වැඩසටහන් වර්ග දෙක

	අගයක් ආපසු දෙන	අගයක් ආපසු නොදෙන
Pascal නම	function	procedure
Python	return අගයක් සහිත def	return අගයක් නැති def
ආපසු දෙන්නේ	එක් අගයක් (උදා ශ්‍රේණියක්)	කිසිවක් නැත – කාර්යයක් පමණක් කරයි
උදාහරණය	GetGrade(82) → 'A'	PrintLine

3. පරාමිති සහ තර්ක

උප වැඩසටහනකට යවන අගයන් **පරාමිති** (parameters) වේ. area(length, width) හි length සහ width පරාමිති වේ. area(5, 4) කැඳවන විට 5 සහ 4 අගයන් **තර්ක** (arguments) වේ.

4. අගයක් ආපසු දෙන ශ්‍රිතයක්

GetGrade ලකුණක් ලබාගෙන O/L ශ්‍රේණි සීමා (A 75, B 65, C 50, S 35) භාවිතයෙන් ශ්‍රේණි අකුරක් **ආපසු දෙයි**.

ව්‍යාජ කේතය (Pseudocode)

```
FUNCTION GetGrade(Marks)
  IF Marks >= 75 THEN RETURN "A"
  ELSE IF Marks >= 65 THEN RETURN "B"
  ELSE IF Marks >= 50 THEN RETURN "C"
  ELSE IF Marks >= 35 THEN RETURN "S"
  ELSE RETURN "W"
  ENDIF
ENDFUNCTION

BEGIN
  FOR k = 1 TO 3
    INPUT M
    DISPLAY GetGrade(M)
  ENDFOR
END
```

අගයක් ආපසු දෙන උප ක්‍රමලේඛය (function): GetGrade

Pascal

```
program GradeFunction;
var
  k, m: integer;

function GetGrade(marks: integer): char;
begin
  if marks >= 75 then
    GetGrade := 'A'
  else if marks >= 65 then
    GetGrade := 'B'
  else if marks >= 50 then
    GetGrade := 'C'
  else if marks >= 35 then
    GetGrade := 'S'
  else
    GetGrade := 'W';
end;

begin
  for k := 1 to 3 do
  begin
    write('Enter mark: ');
    readln(m);
    writeln('Grade: ', GetGrade(m));
  end;
end.
```

Python

```
def get_grade(marks):
    if marks >= 75:
        return "A"
    elif marks >= 65:
        return "B"
    elif marks >= 50:
        return "C"
    elif marks >= 35:
        return "S"
    else:
        return "W"

for k in range(3):
    m = int(input("Enter mark: "))
    print("Grade:", get_grade(m))
```

නියැදි ධාවනය

```
Enter mark: 82
Grade: A
Enter mark: 47
Grade: S
Enter mark: 66
Grade: B
```

> 💡 Pascal හි ප්‍රතිඵලය ලබා දෙන්නේ ශ්‍රිත නමට පැවරීමෙනි: GetGrade := 'A'. Python හි return "A" ලෙස ලියමු.

5. ක්‍රියා පටිපාටි (අගයක් ආපසු නොදෙන)

සාප්පු බිල්පතක PrintLine තුන් වරක් ද එක් එක් අයිතමය සඳහා ShowItem ද භාවිත කරයි. PrintLine ට **පරාමිති නැත**; ShowItem ට තුනක් ඇත.

අගයක් ආපසු නොදෙන උප ක්‍රමලේඛ (procedures): කඩ බිලක්

Pascal

```
program BillProcedures;

procedure PrintLine;
begin
  writeln('------------------------');
end;

procedure ShowItem(item: string; price, qty: integer);
begin
  writeln(item, ': Rs. ', price * qty);
end;

begin
  PrintLine;
  writeln('SAMPLE SHOP BILL');
  PrintLine;
  ShowItem('Bread', 120, 2);
  ShowItem('Milk packet', 280, 1);
  ShowItem('Eggs', 55, 6);
  PrintLine;
end.
```

Python

```
def print_line():
    print("-" * 24)

def show_item(item, price, qty):
    print(item + ": Rs.", price * qty)

print_line()
print("SAMPLE SHOP BILL")
print_line()
show_item("Bread", 120, 2)
show_item("Milk packet", 280, 1)
show_item("Eggs", 55, 6)
print_line()
```

නියැදි ධාවනය

```
------------------------
SAMPLE SHOP BILL
------------------------
Bread: Rs. 240
Milk packet: Rs. 280
Eggs: Rs. 330
------------------------
```

6. ශ්‍රිතයක් බොහෝ වර කැඳවීම

ශ්‍රිතයක් සංඛ්‍යාවක් මෙන්ම ප්‍රකාශනයක් තුළ භාවිත කළ හැක.

පරාමිති දෙකක් සහිත function: සෘජුකෝණාස්‍රයක වර්ගඵලය

Pascal

```
program RectArea;

function Area(length, width: integer): integer;
begin
  Area := length * width;
end;

begin
  writeln('Room 1 = ', Area(5, 4));
  writeln('Room 2 = ', Area(6, 3));
  writeln('Both = ', Area(5, 4) + Area(6, 3));
end.
```

Python

```
def area(length, width):
    return length * width

print("Room 1 =", area(5, 4))
print("Room 2 =", area(6, 3))
print("Both =", area(5, 4) + area(6, 3))
```

නියැදි ධාවනය

```
Room 1 = 20
Room 2 = 18
Both = 38
```

💡 ඔබ දැනටමත් භාවිත කරන ගොඩනැගූ උප වැඩසටහන්: writeln, readln, sqrt (Pascal); print, input, len, round (Python).

ප්‍රධාන පද

උප වැඩසටහන – එක් කාර්යයක් කරන, නමක් ඇති වැඩසටහන් කොටසක්.

ශ්‍රිතය – අගයක් ආපසු දෙන උප වැඩසටහනක්.

ක්‍රියා පටිපාටිය – කාර්යයක් කරන නමුත් අගයක් ආපසු නොදෙන උප වැඩසටහනක්.

කැඳවීම – නම ලිවීමෙන් උප වැඩසටහනක් භාවිත කිරීම.

පරාමිතිය – උප වැඩසටහනකට යවන අගයක් ලබාගන්නා විචල්‍යය.

තර්කය – කැඳවන විට යවන සත්‍ය අගය, උදා area(5, 4) හි 5.

ආපසු අගය – ශ්‍රිතයක් ආපසු දෙන ප්‍රතිඵලය.

නැවත භාවිතය – කේතය එක් වරක් ලියා බොහෝ වර භාවිතය.

කියවීමේ පහසුව – කේතය කියවීමට හා තේරුම් ගැනීමට ඇති පහසුව.

නඩත්තුවේ පහසුව – කේතය නිවැරදි කිරීමට හා වෙනස් කිරීමට ඇති පහසුව.

def – උප වැඩසටහනක් නිර්වචනය කිරීමට Python මූල පදය.

ගොඩනැගූ ශ්‍රිතය – භාෂාව විසින්ම සපයන ශ්‍රිතයක්, උදා len.

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. උප වැඩසටහනක් යනු කුමක්ද?

> **පිළිතුර:** එක් කාර්යයක් කරන, අවශ්‍ය විට කැඳවිය හැකි නමක් ඇති වැඩසටහන් කොටසකි.

2. උප වැඩසටහන්වල වාසි තුනක් දෙන්න.

> **පිළිතුර:** නැවත භාවිතය, කියවීමේ පහසුව සහ නඩත්තුවේ පහසුව.

3. ශ්‍රිතයක් සහ ක්‍රියා පටිපාටියක් අතර වෙනස?

> **පිළිතුර:** ශ්‍රිතයක් අගයක් ආපසු දෙයි; ක්‍රියා පටිපාටියක් එසේ නොකරයි.

4. GetGrade(66) කුමක්ද?

> **පිළිතුර:** B.

5. area(6, 3) ආපසු දෙන්නේ කුමක්ද?

> **පිළිතුර:** 18.

6. area(5, 4) හි 5 සහ 4 යනු මොනවාද?

> **පිළිතුර:** තර්ක (length සහ width පරාමිති සඳහා අගයන්).

බහුවරණ ප්‍රශ්න

1. එක් කාර්යයක් කරන නමක් ඇති වැඩසටහන් කොටසක්...

A. දත්ත වර්ගයකි
B. විවරණයකි
C. නියතයකි
D. උප වැඩසටහනකි

2. අගයක් ආපසු දෙන උප වැඩසටහනක්...

A. ශ්‍රිතයකි
B. අරාවකි
C. ක්‍රියා පටිපාටියකි
D. ලූපයකි

3. Pascal හි අගයක් ආපසු නොදෙන උප වැඩසටහනක්...

A. record
B. const
C. procedure
D. function

4. “එක් වරක් ලියන්න, බොහෝ වර භාවිත කරන්න” විස්තර කරන්නේ...

A. කැදැලි කිරීම
B. සම්පාදනය
C. නැවත භාවිතය
D. දෝෂ නිරාකරණය

5. කාර්යයක් එක් තැනක පමණක් නිවැරදි කිරීම පෙන්වන්නේ...

A. නඩත්තුවේ පහසුව
B. පුනරාවර්තනය
C. සෘජු ප්‍රවේශය
D. ප්‍රමුඛතාව

6. PrintLine වැනි පැහැදිලි නම් වැඩි දියුණු කරන්නේ...

A. කියවීමේ පහසුව
B. මුද්‍රකයේ වේගය
C. දත්ත වර්ගය
D. මතක ප්‍රමාණය

7. GetGrade(82) ආපසු දෙන්නේ...

A. S
B. B
C. C
D. A

8. GetGrade(47) ආපසු දෙන්නේ...

A. C
B. S
C. W
D. B

9. area(5, 4) + area(6, 3) සමාන වන්නේ...

A. 18
B. 20
C. 9
D. 38

10. def area(length, width): හි length සහ width...

A. වෙන් කළ වචන
B. පරාමිති
C. අරා
D. නියත

11. උප වැඩසටහනක් නිර්වචනය කරන Python මූල පදය...

A. func
B. proc
C. def
D. sub

12. Python හි ශ්‍රිතයක් ප්‍රතිඵලය ආපසු දෙන්නේ...

A. break
B. return
C. input
D. print

13. Pascal හි ශ්‍රිතයක් අගයක් ආපසු දෙන්නේ...

A. ශ්‍රිත නමට පැවරීමෙන්
B. print භාවිතයෙන්
C. end. ලිවීමෙන්
D. readln භාවිතයෙන්

14. ShowItem('Eggs', 55, 6) පෙන්වන්නේ...

A. Eggs: Rs. 55
B. Eggs: Rs. 330
C. Eggs: Rs. 6
D. Eggs: Rs. 61

15. ගොඩනැගූ Python ශ්‍රිතයක් කුමක්ද?

A. PrintLine
B. ShowItem
C. len
D. GetGrade

ව්‍යුහගත ප්‍රශ්න

1. උප වැඩසටහන් භාවිතයේ වාසි තුනක් සඳහන් කරන්න. (ලකුණු 3) *(ලකුණු 3)*

> නැවත භාවිතය – එක් වරක් ලියා බොහෝ වර කැඳවීම; කියවීමේ පහසුව – තේරුම් ගැනීමට පහසු; නඩත්තුව – එක් තැනක වෙනස් කිරීම (එකට 1).

2. අගයක් ආපසු දෙන සහ ආපසු නොදෙන උප වැඩසටහන් එක් එකකට උදාහරණයක් සමඟ වෙන්කර දක්වන්න. (ලකුණු 4) *(ලකුණු 4)*

> අගයක් ආපසු දෙන (ශ්‍රිතය) අගයක් ලබා දෙයි, උදා GetGrade (2); ආපසු නොදෙන (ක්‍රියා පටිපාටිය) කාර්යයක් පමණක් කරයි, උදා PrintLine (2).

3. ලකුණක් ලබාගෙන O/L ශ්‍රේණිය A, B, C, S හෝ W ආපසු දෙන ශ්‍රිතයක් (Pascal හෝ Python) ලියන්න. (ලකුණු 5) *(ලකුණු 5)*

පරාමිතිය සහිත ශ්‍රිත ශීර්ෂය (1); IF ≥ 75 A (1); ELSE IF ≥ 65 B, ≥ 50 C (1); ≥ 35 S නැතහොත් W (1); ප්‍රතිඵලය return / පැවරීම (1).

අගයක් ආපසු දෙන උප ක්‍රමලේඛය (function): GetGrade

Pascal

```
program GradeFunction;
var
  k, m: integer;

function GetGrade(marks: integer): char;
begin
  if marks >= 75 then
    GetGrade := 'A'
  else if marks >= 65 then
    GetGrade := 'B'
  else if marks >= 50 then
    GetGrade := 'C'
  else if marks >= 35 then
    GetGrade := 'S'
  else
    GetGrade := 'W';
end;

begin
  for k := 1 to 3 do
  begin
    write('Enter mark: ');
    readln(m);
    writeln('Grade: ', GetGrade(m));
  end;
end.
```

Python

```
def get_grade(marks):
    if marks >= 75:
        return "A"
    elif marks >= 65:
        return "B"
    elif marks >= 50:
        return "C"
    elif marks >= 35:
        return "S"
    else:
        return "W"

for k in range(3):
    m = int(input("Enter mark: "))
    print("Grade:", get_grade(m))
```

නියැදි ධාවනය

```
Enter mark: 82
Grade: A
Enter mark: 47
Grade: S
Enter mark: 66
Grade: B
```

**4. def area(length, width): return length * width
print(area(5, 4) + area(6, 3)) මගින් පෙන්වන්නේ කුමක්ද? පරාමිති සහ තර්ක නම් කරන්න. (ලකුණු 3)** *(ලකුණු 3)*

> 38 (1); පරාමිති: length, width (1); තර්ක: 5, 4, 6, 3 (1).

5. අයිතම නමක්, මිලක් සහ ප්‍රමාණයක් ලබාගෙන අයිතම මුළු මුදල පෙන්වන ShowItem ක්‍රියා පටිපාටියක් ලියන්න. එක් කැඳවීමක් පෙන්වන්න. (ලකුණු 4) *(ලකුණු 4)*

පරාමිති තුනක් සහිත ශීර්ෂය (1); price × qty ගණනය (1); පෙන්වීම (1); කැඳවීම උදා ShowItem('Bread', 120, 2) (1).

අගයක් ආපසු නොදෙන උප ක්‍රමලේඛ (procedures): කඩ බිලක්

Pascal

```
program BillProcedures;

procedure PrintLine;
begin
  writeln('------------------------');
end;

procedure ShowItem(item: string; price, qty: integer);
begin
  writeln(item, ': Rs. ', price * qty);
end;

begin
  PrintLine;
  writeln('SAMPLE SHOP BILL');
  PrintLine;
  ShowItem('Bread', 120, 2);
  ShowItem('Milk packet', 280, 1);
  ShowItem('Eggs', 55, 6);
  PrintLine;
end.
```

Python

```
def print_line():
    print("-" * 24)

def show_item(item, price, qty):
    print(item + ": Rs.", price * qty)

print_line()
print("SAMPLE SHOP BILL")
print_line()
show_item("Bread", 120, 2)
show_item("Milk packet", 280, 1)
show_item("Eggs", 55, 6)
print_line()
```

නියැදි ධාවනය

```
------------------------
SAMPLE SHOP BILL
------------------------
Bread: Rs. 240
Milk packet: Rs. 280
Eggs: Rs. 330
------------------------
```

සාරාංශය

- උප වැඩසටහන = එක් කාර්යයක් කරන නමක් ඇති කොටස.
- වාසි: නැවත භාවිතය, කියවීමේ පහසුව, නඩත්තුව.
- ශ්‍රිතය අගයක් ආපසු දෙයි.
- ක්‍රියා පටිපාටිය අගයක් ආපසු නොදෙයි.
- පරාමිති ලබා ගනී; තර්ක යවයි.
- Python: def … return.
- Pascal: FunctionName := value.
- ගොඩනැගූ: print, len, writeln, sqrt.

MCQ පිළිතුරු – 1.10

1. **D** – උප වැඩසටහන.
2. **A** – ශ්‍රිතය.
3. **C** – Procedure.
4. **C** – නැවත භාවිතය.
5. **A** – නඩත්තුව.
6. **A** – කියවීමේ පහසුව.
7. **D** – 82 ≥ 75.
8. **B** – 35 ≤ 47 < 50.
9. **D** – 20 + 18.
10. **B** – පරාමිති.
11. **C** – def.
12. **B** – return.
13. **A** – GetGrade := 'A'.
14. **B** – 55 × 6.
15. **C** – ගොඩනැගූ.

© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.

G.C.E. O/L ICT · 11 ශ්‍රේණිය · පාඩම 1

1.11 ක්‍රමලේඛන භාෂා, පරිවර්තක සහ දෝෂ නිරාකරණය

විෂය නිර්දේශය 10.11. පරිගණකයක් තේරුම් ගන්නේ **0 සහ 1** පමණි. මිනිසුන් print, if වැනි වචන කැමතියි. මෙම කොටසේ විවිධ **ක්‍රමලේඛන භාෂා වර්ග**, **පරිවර්තකයක්** (අර්ථවිනයක හෝ සම්පාදකයක) අපගේ කේතය යන්ත්‍ර කේතයට හරවන ආකාරය සහ වැඩසටහනක් **පරීක්ෂා කර දෝෂ නිරාකරණය** කරන ආකාරය බලමු.

1. පහළ මට්ටමේ සහ ඉහළ මට්ටමේ භාෂා

වර්ගය	පෙනුම	සටහන්
යන්ත්‍ර භාෂාව	10110000 01100001	0 සහ 1 පමණි. CPU එක කෙළින්ම ධාවනය කරයි. මිනිසුන්ට ඉතා අපහසුයි.
ඇසෙම්බ්ලි භාෂාව	MOV AL, 61h	කෙටි වචන (mnemonics). **ඇසෙම්බ්ලරයක්** අවශ්‍යයි. CPU එක මත රඳා පවතී.
ඉහළ මට්ටමේ භාෂාව	print("Hello")	ඉංග්‍රීසි වැනි, ඉගෙනීමට පහසු, බොහෝ පරිගණකවල ධාවනය වේ. සම්පාදකයක් හෝ අර්ථවිනයකයක් අවශ්‍යයි. උදා Python, Pascal, Java, C.

යන්ත්‍ර සහ ඇසෙම්බ්ලි භාෂා දෘඪාංගයට සමීප නිසා **පහළ මට්ටමේ** භාෂා ලෙස හැඳින්වේ.

2. ඉහළ මට්ටමේ භාෂා වර්ග

ක්‍රියා පටිපාටික
එය කරන්නේ **කෙසේද** යන්න පියවරෙන් පියවර පරිගණකයට කියන්න. උදා Pascal, C.

ප්‍රකාශන
ඔබට අවශ්‍ය **දේ** කියන්න; ක්‍රමය එය සොයා ගනී. උදා SQL, Prolog.

ව්‍යුහගත
අනුක්‍රමය, වරණය, පුනරාවර්තනය සහ උප වැඩසටහන් වලින් පමණක් ගොඩනැගේ. උදා Pascal.

වස්තු-නැඹුරු
දත්ත සහ ක්‍රියා එකට තබන **වස්තු** වලින් ගොඩනැගේ (උදා Student වස්තුවක්). උදා Java, C++, Python.

3. ක්‍රමලේඛන සහ ස්ක්‍රිප්ටින් භාෂා

ස්ක්‍රිප්ටින් භාෂා සාමාන්‍යයෙන් අර්ථවිනයනය කරන අතර වෙනත් මෘදුකාංග පාලනයට හෝ වෙබ් පිටුවලට ක්‍රියා එක් කිරීමට භාවිත වේ – උදා **JavaScript** (වෙබ් පිටු), **PHP** (වෙබ් සේවාදායක), Python ස්ක්‍රිප්ට්. C හෝ Pascal වැනි **ක්‍රමලේඛන භාෂා** සාමාන්‍යයෙන් ස්වාධීන වැඩසටහන් ලෙස සම්පාදනය කෙරේ.

2026 නව 2026 – **Python** දැන් ලෝකයේ වඩාත්ම ජනප්‍රිය ක්‍රමලේඛන භාෂාවලින් එකකි. කේතය කෙටි සහ කියවීමට පහසු නිසා එය AI, දත්ත විද්‍යාව, වෙබ් අඩවි සහ පාසල් ඉගැන්වීම සඳහා භාවිත වේ.

4. පරිවර්තක: සම්පාදකය සහ අර්ථවිනයකය

	සම්පාදකය	**අර්ථවිනයකය**
පරිවර්තනය කරන ආකාරය	සම්පූර්ණ වැඩසටහනම එකවර	වරකට එක් පේළියක්, පසුව ධාවනය
ප්‍රතිදානය	ක්‍රියාත්මක ගොනුවක් (උදා .exe)	වෙනම ගොනුවක් නැත
දෝෂ	සම්පූර්ණ සම්පාදනයෙන් පසු ලැයිස්තුගත වේ	දෝෂයක් ඇති පළමු පේළියේ නතර වේ
ධාවන වේගය	වේගවත්	මන්දගාමී
උදාහරණය	Free Pascal (FPC), C	Python

අප ලියන කේතය **මූලාශ්‍ර කේතයයි** (source code). පරිවර්තිත ප්‍රතිඵලය **යන්ත්‍ර කේතයයි** (object code). **ඇසෙම්බ්ලරයක්** ඇසෙම්බ්ලි භාෂාව පරිවර්තනය කරයි.

5. වැඩසටහන් දෝෂ

දෝෂය	**තේරුම**	**උදාහරණය**
වාක්‍ය රීති දෝෂය	භාෂාවේ ව්‍යාකරණ රීති කැඩීම. පරිවර්තකය එය සොයා ගනී.	prnt("Hi") / missing ;
ධාවන කාල දෝෂය	ධාවනය වන අතරතුර වැඩසටහන නතර වේ.	බිංදුවෙන් බෙදීම
තාර්කික දෝෂය	වැඩසටහන ධාවනය වේ නමුත් **වැරදි පිළිතුරක්** දෙයි.	total / 3 වෙනුවට total / 2

6. පරීක්ෂාව සහ දෝෂ නිරාකරණය

පරීක්ෂාව යනු තෝරාගත් දත්ත සමඟ වැඩසටහන ධාවනය කර ප්‍රතිදානය අපේක්ෂිත පිළිතුර සමඟ සැසඳීමයි. **දෝෂ නිරාකරණය** (debugging) යනු දෝෂ (bugs) සොයා නිවැරදි කිරීමයි. පහත වැඩසටහන ලකුණු තුනක සාමාන්‍යය සොයා ගත යුතු නමුත් එහි **තාර්කික දෝෂයක්** ඇත.

තාර්කික දෝෂය: වැරදි සංඛ්‍යාවෙන් බෙදීම

Pascal

```
program AverageBug;
var
  m1, m2, m3, total: integer;
  avg: real;
begin
  write('Enter mark 1: ');
  readln(m1);
  write('Enter mark 2: ');
  readln(m2);
  write('Enter mark 3: ');
  readln(m3);
  total := m1 + m2 + m3;
  avg := total / 2;            { BUG: should be / 3 }
  writeln('Average = ', avg:0:2);
end.
```

Python

```
m1 = int(input("Enter mark 1: "))
m2 = int(input("Enter mark 2: "))
m3 = int(input("Enter mark 3: "))
total = m1 + m2 + m3
avg = total / 2               # BUG: should be / 3
print(f"Average = {avg:.2f}")
```

නියැදි ධාවනය

```
Enter mark 1: 75
Enter mark 2: 68
Enter mark 3: 72
Average = 107.50
```

💡 75, 68 සහ 72 හි අපේක්ෂිත සාමාන්‍යය = 215 ÷ 3 = 71.67, නමුත් වැඩසටහන 107.50 පෙන්වයි. නිවැරදි කිරීම: 3 න් බෙදන්න. **සාමාන්‍ය** (60), **මායිම්** (0, 100, 35) සහ **අවලංගු** (−5, 150) දත්ත සමඟ පරීක්ෂා කරන්න.

දෝෂ නිරාකරණ ක්‍රම: දෝෂ පණිවිඩය කියවන්න, **අනුරේඛන වගුවක්** (dry run) භාවිත කරන්න, විචල්‍ය අගයන් මුද්‍රණය කරන්න, සහ IDE එකක debugger එක භාවිතයෙන් **පියවරෙන් පියවර** ධාවනය කරන්න.

7. 2026 IDE සහ මෙවලම්

IDE එකක් (ඒකාබද්ධ සංවර්ධන පරිසරය) සංස්කාරකයක්, පරිවර්තකයක් සහ debugger එකක් එක් වැඩසටහනක ලබා දෙයි.

- **IDLE** – Python සමඟ නොමිලේ ලැබේ; ආරම්භකයින්ට සරලයි.
- **Thonny** – විචල්‍ය පියවරෙන් පියවර පෙන්වන ආරම්භක Python IDE එකකි.
- **VS Code** – Python, Pascal සහ බොහෝ භාෂා සඳහා ජනප්‍රිය නොමිලේ සංස්කාරකයකි.

- **Free Pascal / Lazarus** – Pascal සඳහා; **ඔන්ලයින් සම්පාදක** වෙබ් බ්‍රවුසරයක කේතය ධාවනය කරයි.
- **Scratch** – බ්ලොක් කේතනය: ටයිප් කිරීම වෙනුවට බ්ලොක් ඇදගෙන යන්න; Python ට පෙර විනෝදජනක පළමු පියවරකි.

2026 නව 2026 – **AI සහායක** (VS Code තුළ ඇති chatbot වැනි) දෝෂ පැහැදිලි කර කේත යෝජනා කළ හැක. ඒවා **වගකීමෙන්** භාවිත කරන්න: මුලින් ගැටලුව ඔබම උත්සාහ කරන්න, භාවිත කිරීමට පෙර සෑම පේළියක්ම තේරුම් ගන්න, කේතය පරීක්ෂා කරන්න, විභාග හෝ පැවරුම්වල AI වැඩ ඔබේ ලෙස ඉදිරිපත් නොකරන්න, සහ පුද්ගලික දත්ත ඒවාට ඇතුළත් නොකරන්න.

ප්‍රධාන පද

යන්ත්‍ර භාෂාව – CPU එක කෙළින්ම ධාවනය කරන 0 සහ 1 උපදෙස්.

ඇසෙම්බ්ලි භාෂාව – MOV වැනි කෙටි වචන භාවිත කරන පහළ මට්ටමේ භාෂාව.

ඉහළ මට්ටමේ භාෂාව – Python හෝ Pascal වැනි ඉංග්‍රීසි වැනි භාෂාව.

සම්පාදකය – සම්පූර්ණ වැඩසටහනම එකවර යන්ත්‍ර කේතයට පරිවර්තනය කරයි.

අර්ථවිනයකය – වරකට එක් පේළියක් පරිවර්තනය කර ධාවනය කරයි.

ඇසෙම්බ්ලරය – ඇසෙම්බ්ලි භාෂාව යන්ත්‍ර කේතයට පරිවර්තනය කරයි.

මූලාශ්‍ර කේතය – ක්‍රමලේඛකයා ලියූ වැඩසටහන.

වාක්‍ය රීති දෝෂය – භාෂාවේ ව්‍යාකරණ රීතිවල වැරැද්දක්.

තාර්කික දෝෂය – වැඩසටහන ධාවනය වුවත් වැරදි ප්‍රතිඵලයක් දෙයි.

ධාවන කාල දෝෂය – ධාවනය වන අතරතුර වැඩසටහන නවත්වන දෝෂයක්.

දෝෂ නිරාකරණය – වැඩසටහනක දෝෂ සොයා නිවැරදි කිරීම.

IDE – සංස්කාරකය, පරිවර්තකය සහ debugger එක එකට ඇති වැඩසටහනක්.

ආදර්ශ ප්‍රශ්න සහ පිළිතුරු

1. ඉහළ මට්ටමේ වැඩසටහන්වලට පරිවර්තකයක් අවශ්‍ය ඇයි?

පිළිතුර: CPU එක තේරුම් ගන්නේ යන්ත්‍ර කේතය (0 සහ 1) පමණි.

2. සම්පාදකයක් සහ අර්ථවිනයකයක් අතර එක් වෙනසක් දෙන්න.

පිළිතුර: සම්පාදකය සම්පූර්ණ වැඩසටහනම එකවර පරිවර්තනය කරයි; අර්ථවිනයකය වරකට එක් පේළියක්.

3. ක්‍රියා පටිපාටික සහ ප්‍රකාශන?

පිළිතුර: ක්‍රියා පටිපාටික කෙසේද (පියවර) කියයි; ප්‍රකාශන අවශ්‍ය දේ කියයි.

4. total / 3 වෙනුවට total / 2 කුමන දෝෂයක්ද?

> **පිළිතුර:** තාර්කික දෝෂයකි.

5. Python සඳහා IDE දෙකක් නම් කරන්න.

> **පිළිතුර:** IDLE සහ Thonny (හෝ VS Code).

6. AI සහායක වගකීමෙන් භාවිතයට එක් රීතියක් දෙන්න.

> **පිළිතුර:** කේතය තේරුම් ගෙන පරීක්ෂා කරන්න; එය ඔබේ වැඩ ලෙස ඉදිරිපත් නොකරන්න.

බහුවරණ ප්‍රශ්න

1. CPU එක කෙළින්ම ධාවනය කරන භාෂාව...

A. Python
B. SQL
C. යන්ත්‍ර භාෂාව
D. Pascal

2. MOV AL, 61h ලියා ඇත්තේ...

A. ඇසෙම්බ්ලි භාෂාවෙනි
B. SQL වලිනි
C. Python වලිනි
D. HTML වලිනි

3. ඇසෙම්බ්ලි භාෂාව පරිවර්තනය කරන්නේ...

A. සංස්කාරකයකි
B. අර්ථවිනයකයක් පමණි
C. ඇසෙම්බ්ලරයකි
D. බ්‍රවුසරයකි

4. ඉහළ මට්ටමේ භාෂාවක් කුමක්ද?

A. Python
B. යන්ත්‍ර කේතය
C. ද්විමය
D. ඇසෙම්බ්ලි

5. සම්පාදකයක් පරිවර්තනය කරන්නේ...

A. දත්ත පමණි
B. සම්පූර්ණ වැඩසටහනම එකවර
C. වරකට එක් පේළියක්
D. විවරණ පමණි

6. Python සාමාන්‍යයෙන් ධාවනය කරන්නේ...

A. අර්ථවිනයකයක්
B. ස්කෑනරයක්
C. මුද්‍රකයක්
D. ඇසෙම්බ්ලරයක්

7. අවශ්‍ය දත්ත මොනවාදැයි කියන SQL...

A. යන්ත්‍ර කේතයකි
B. ප්‍රකාශන භාෂාවකි
C. ඇසෙම්බ්ලි භාෂාවකි
D. ක්‍රියා පටිපාටික භාෂාවකි

8. දත්ත සහ ක්‍රියා තබන වස්තු වටා ගොඩනැගූ භාෂාවක්...

A. යන්ත්‍ර
B. වස්තු-නැඹුරු
C. ඇසෙම්බ්ලි
D. ප්‍රකාශන

9. වෙබ් පිටුවලට ක්‍රියා එක් කිරීමට භාවිත වන JavaScript...

A. ඇසෙම්බ්ලරයකි
B. යන්ත්‍ර භාෂාවකි
C. දත්ත වර්ගයකි
D. ස්ක්‍රිප්ටින් භාෂාවකි

10. Python හි prnt("Hi") ලිවීමෙන් ඇති වන්නේ...

A. දෘඪාංග දෝෂයකි
B. වාක්‍ය රීති දෝෂයකි
C. තාර්කික දෝෂයකි
D. දෝෂයක් නැත

11. ධාවනය වුවත් වැරදි පිළිතුරක් දෙන වැඩසටහනක ඇත්තේ...

A. වෛරසයකි
B. වාක්‍ය රීති දෝෂයකි
C. සම්පාදකයකි
D. තාර්කික දෝෂයකි

12. ධාවනයේදී බිංදුවෙන් බෙදීම නිසා ඇති වන්නේ...

A. ධාවන කාල දෝෂයකි
B. වාක්‍ය රීති දෝෂයකි
C. විවරණයකි
D. තාර්කික දෝෂයක් පමණි

13. ලකුණු 75, 68, 72 total / 2 සමඟ Average = ...

A. 71.67
B. 72.50
C. 107.50
D. 215.00

14. වැඩසටහනක දෝෂ සොයා නිවැරදි කිරීම...

A. ප්‍රකාශ කිරීමයි
B. සම්පාදනයයි
C. කැදැලි කිරීමයි
D. දෝෂ නිරාකරණයයි

15. AI කේත සහායකයක් වගකීමෙන් භාවිත කිරීමේ ක්‍රමයක්...

A. එහි පිළිතුරු ඔබේ වැඩ ලෙස ඉදිරිපත් කිරීම
B. ඔබේ මුරපද එයට ඇතුළත් කිරීම
C. භාවිතයට පෙර එහි කේතය තේරුම් ගෙන පරීක්ෂා කිරීම
D. කේතය කිසිදා පරීක්ෂා නොකිරීම

ව්‍යුහගත ප්‍රශ්න

1. (i) පහළ මට්ටමේ භාෂා දෙකක් නම් කරන්න. (ලකුණු 1)
(ii) ඉහළ මට්ටමේ භාෂාවල වාසි දෙකක් දෙන්න. (ලකුණු 2) *(ලකුණු 3)*

(i) යන්ත්‍ර භාෂාව, ඇසෙම්බ්ලි භාෂාව. (ii) ඉංග්‍රීසි වැනි / ඉගෙනීමට පහසු; විවිධ පරිගණකවල ධාවනය; අඩු දෝෂ (ඕනෑම දෙකක්).

2. සම්පාදකයක් සහ අර්ථවිනයකයක් ශීර්ෂ තුනක් යටතේ සසඳන්න. (ලකුණු 3) *(ලකුණු 3)*

පරිවර්තනය: සම්පූර්ණ වැඩසටහන එදිරිව පේළියෙන් පේළිය; ප්‍රතිදානය: ක්‍රියාත්මක ගොනුව එදිරිව නැත; දෝෂ: එකට එදිරිව පළමු එකේ නතර; වේගය: වේගවත් එදිරිව මන්දගාමී (ඕනෑම තුනක්).

3. (a) ක්‍රියා පටිපාටික සහ ප්‍රකාශන භාෂා (b) ව්‍යුහගත සහ වස්තු-නැඹුරු ක්‍රමලේඛනය අතර වෙනස සඳහන් කරන්න. (ලකුණු 4) *(ලකුණු 4)*

(a) ක්‍රියා පටිපාටික පියවරෙන් පියවර කෙසේද දෙයි (Pascal); ප්‍රකාශන අවශ්‍ය දේ සඳහන් කරයි (SQL) (2). (b) ව්‍යුහගත අනුක්‍රමය, වරණය, පුනරාවර්තනය සහ උප වැඩසටහන් භාවිත කරයි; OOP දත්ත සහ ක්‍රියා තබන වස්තු වලින් ගොඩනගයි (2).

4. ලකුණු තුනක සාමාන්‍යය සොයන වැඩසටහනක් 75, 68, 72 සඳහා 107.50 පෙන්වයි.
(i) මෙය කුමන දෝෂයක්ද? (ලකුණු 1)
(ii) නිවැරදි පිළිතුර කුමක්ද, එය නිවැරදි කරන්නේ කෙසේද? (ලකුණු 2)
(iii) වෙනත් දෝෂ වර්ග දෙකක් උදාහරණ සමඟ දෙන්න. (ලකුණු 2) *(ලකුණු 5)*

(i) තාර්කික දෝෂය. (ii) 71.67; එකතුව 3 න් බෙදන්න. (iii) වාක්‍ය රීති දෝෂය – prnt("Hi"); ධාවන කාල දෝෂය – බිංදුවෙන් බෙදීම.

තාර්කික දෝෂය: වැරදි සංඛ්‍යාවෙන් බෙදීම

Pascal

```
program AverageBug;
var
  m1, m2, m3, total: integer;
  avg: real;
begin
  write('Enter mark 1: ');
  readln(m1);
  write('Enter mark 2: ');
  readln(m2);
  write('Enter mark 3: ');
  readln(m3);
  total := m1 + m2 + m3;
  avg := total / 2;            { BUG: should be / 3 }
  writeln('Average = ', avg:0:2);
end.
```

Python

```
m1 = int(input("Enter mark 1: "))
m2 = int(input("Enter mark 2: "))
m3 = int(input("Enter mark 3: "))
total = m1 + m2 + m3
avg = total / 2               # BUG: should be / 3
print(f"Average = {avg:.2f}")
```

නියැදි ධාවනය

```
Enter mark 1: 75
Enter mark 2: 68
Enter mark 3: 72
Average = 107.50
```

5. IDE එකක් යනු කුමක්ද? Python සඳහා භාවිත වන IDE දෙකක් නම් කර AI සහායක වගකීමෙන් භාවිතයට රීති දෙකක් සඳහන් කරන්න. (ලකුණු 4) *(ලකුණු 4)*

සංස්කාරකය, පරිවර්තකය සහ debugger එක එකම වැඩසටහනක (1); IDLE, Thonny හෝ VS Code (1); මුලින් ඔබම උත්සාහ කිරීම / කේතය තේරුම් ගෙන පරීක්ෂා කිරීම / ඔබේ ලෙස ඉදිරිපත් නොකිරීම / පුද්ගලික දත්ත නොදීම (2).

සාරාංශය

- පහළ මට්ටම: යන්ත්‍ර (0/1) සහ ඇසෙම්බ්ලි.
- ඉහළ මට්ටම: ඉංග්‍රීසි වැනි – Python, Pascal.
- ක්‍රියා පටිපාටික = කෙසේද; ප්‍රකාශන = කුමක්ද.
- ව්‍යුහගත එදිරිව වස්තු-නැඹුරු.
- සම්පාදකය: සම්පූර්ණ වැඩසටහන; අර්ථවිනයකය: පේළියෙන් පේළිය.
- දෝෂ: වාක්‍ය රීති, ධාවන කාල, තාර්කික.
- සාමාන්‍ය, මායිම්, අවලංගු දත්ත සමඟ පරීක්ෂා කරන්න.
- IDE: IDLE, Thonny, VS Code; AI වගකීමෙන් භාවිත කරන්න.

MCQ පිළිතුරු – 1.11

1. **C** – 0 සහ 1.
2. **A** – Mnemonics.
3. **C** – ඇසෙම්බ්ලරය.
4. **A** – ඉංග්‍රීසි වැනි.
5. **B** – සම්පාදකය.
6. **A** – අර්ථවිනයකය.
7. **B** – කෙසේද නොව කුමක්ද.
8. **B** – OOP.
9. **D** – ස්ක්‍රිප්ටින්.
10. **B** – වැරදි අක්ෂර වින්‍යාසය.
11. **D** – තාර්කික.
12. **A** – ධාවන කාල.
13. **C** – 215 ÷ 2.
14. **D** – දෝෂ නිරාකරණය.
15. **C** – වගකීම් සහගත භාවිතය.

© 2026 schoolict.net (Ruwan Mangala Suraweera). සියලු හිමිකම් ඇවිරිණි.