1 ගැටලු විසඳීමට ක්‍රමලේඛ ලිවීම

මෙම පාඩම හැදෑරීමෙන් ඔබට,

- ගැටලුවක් විශ්ලේෂණය කිරීම හා ගැටලුවක් සඳහා ඇල්ගොරිතම ගොඩ නැගීම
- පාලන ව්‍යුහය හා ඒවායේ අවශ්‍යතාව
- ගැලීම් සටහන් ඇඳීම, ව්‍යාජ කේත ලිවීම සහ ඒවා අතර පරිවර්තන කිරීම
- ගැටලුවක් සඳහා විකල්ප විසඳුම් සෙවීම
- විචල්‍ය හා නියත
- පැස්කල් ක්‍රමලේඛන භාෂාව
- ක්‍රමලේඛ භාෂාවල පරිණාමය

පිළිබඳ ව අවබෝධයක් ලබා ගැනීමට හැකි වනු ඇත.

1.1 ගැටලුවක් විශ්ලේෂණය කිරීම

ගැටලුවක් විසඳීම සඳහා යොදාගන්නා අමුද්‍රව්‍ය ආදාන (Input) ලෙස හැඳින්වේ. එය විසඳීමෙන් ලබා ගන්නා ප්‍රතිඵලය, ප්‍රතිදානය (Output) ලෙස හැඳින්වේ. ආදාන, ප්‍රතිදාන බවට පත්කිරීම, ක්‍රියාවලිය හෙවත් සැකසීම (Process) ලෙස හැඳින්වේ. ක්‍රියාවලියක් පියවරෙන් පියවර සිදු වන අතර ඒවා අනුපිළිවෙළකින් දැක්වීම ඉතා වැදගත් වේ. ගැටලුවක් විශ්ලේෂණය කිරීමේ දී එහි ආදාන, ක්‍රියාවලිය හා ප්‍රතිදාන වෙන් වෙන් ව හඳුනා ගනු ලබයි.

උදාහරණ -

ගැටලුව 1 : තැපැල් කිරීමට සුදුසු ලිපියක් සෑදීම

ආදාන : ලිපිය ලිවීමට සුදුසු කොළයක් සහ පෑනක්
ලියුම් කවරයක් සහ මුද්දර
මැලියම්

ක්‍රියාවලිය :
1. ලිපිය ලිවීම
2. ලිපිය නවා ලියුම් කවරයට බහා ලීම
3. ලියුම් කවරය ඇලවීම
4. යා යුතු ලිපිනය ලියුම් කවරයේ ලිවීම
5. මුද්දර ඇලවීම

ප්‍රතිදානය : තැපැල් කිරීමට සුදුසු තත්ත්වයේ පවතින ලිපියක්

සටහන : මෙම ක්‍රියාවලියේ පියවර අංක 4 සහ 5 හුවමාරු කර සිදු කළ හැකි ය. එහෙත් අනෙකුත් පියවර අනුපිළිවෙළින් ම සිදු කළ යුතු වේ.

ගැටලුව 2 : රසවත් තේ කෝප්පයක් පිළියෙල කිරීම

ආදාන : තේ කොළ, සීනි, උණු ජලය

ක්‍රියාවලිය :
1. තේ කොළ පෙරනයට දැමීම
2. පෙරනය තුළින් උණු ජලය කෝප්පයට දැමීම
3. සීනි ටිකක් තේ කෝප්පයට දැමීම
4. හැන්දෙන් තේ කෝප්පයෙහි වූ සීනි දිය කිරීම
5. හැන්දෙන් තේ ස්වල්පයක් ගෙන රසය පරීක්ෂා කිරීම
6. රසය ප්‍රමාණවත් නොවේ නම් පියවර අංක 3ට යාම සහ 4 සහ 5 පියවර නැවත සිදු කිරීම.

ප්‍රතිදානය : රසවත් තේ කෝප්පය

ගැටලුව 3 : පොත් පාර්සලයක ඇති පිටු 40 සහ පිටු 80 පොත් පිළිවෙළින් සඳුනි සහ අනූපම යන සහෝදරයන් දෙදෙනා අතර බෙදීම

ආදානය : පොත් පාර්සලය

ක්‍රියාවලිය :
1. පොත් පාර්සලය විවෘත කිරීම
2. පාර්සලයෙන් පොතක් ගැනීම
3. පිටු සංඛ්‍යාව 40 නම් සඳුනිට දීම
4. පිටු සංඛ්‍යාව 80 නම් අනූපමට දීම
5. පාර්සලයේ ඇති පොත් අවසන් වන තෙක් පියවර අංක 2 වෙත යාම

ප්‍රතිදානය : සඳුනිට පිටු 40 පොත් ලැබීම
අනූපමට පිටු 80 පොත් ලැබීම

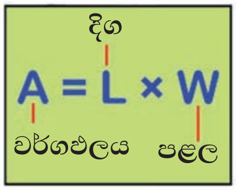
ගැටලුව 4 : සංඛ්‍යා දෙකක එකතුව සෙවීම

ආදාන : සංඛ්‍යා දෙක

ක්‍රියාවලිය : සංඛ්‍යා දෙක එකතු කිරීම

ප්‍රතිදානය : එකතුව

ගැටලුව 5 : ඍජුකෝණාස්‍රයක වර්ගඵලය සෙවීම



ආදානය : ඍජුකෝණාස්‍රයේ දිග හා පළල

ක්‍රියාවලිය : වර්ගඵලය = දිග × පළල

ප්‍රතිදානය : වර්ගඵලය

ගැටලුව 6 : සංඛ්‍යා දෙකකින් විශාල සංඛ්‍යාව සෙවීම

ආදානය : සංඛ්‍යා දෙක

ක්‍රියාවලිය : සංඛ්‍යා දෙක සැසඳීමෙන් විශාල සංඛ්‍යාව අනාවරණය කිරීම

ප්‍රතිදානය : විශාල සංඛ්‍යාව

ගැටලුව 7 : සංඛ්‍යාවක් ඔත්තේ ද ඉරට්ටේ ද යන්න සෙවීම

ආදානය : සංඛ්‍යාව

ක්‍රියාවලිය : සංඛ්‍යාව දෙකෙන් බෙදා ශේෂය සෙවීම

ශේෂය = 0 නම් ඉරට්ටේ සංඛ්‍යාවක් බව තීරණය කිරීම

ශේෂය = 1 නම් ඔත්තේ සංඛ්‍යාවක් බව තීරණය කිරීම

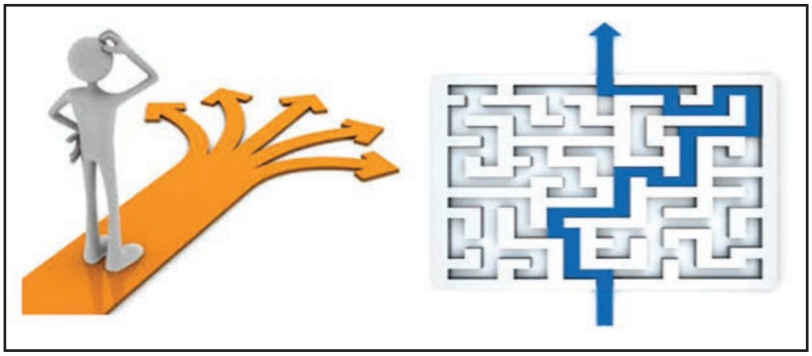
ප්‍රතිදානය : සංඛ්‍යාව ඔත්තේ හෝ ඉරට්ටේ බව දැක්වීම

ක්‍රියාකාරකම

1. ටොෆි 100ක් 20 දෙනෙකු අතරේ සම සම ව බෙදීම සම්බන්ධයෙන් ආදානය, ක්‍රියාවලිය සහ ප්‍රතිදානය ලියා දක්වන්න.
2. සරුංගලයක් සෑදීම සඳහා ආදානය, ක්‍රියාවලිය සහ ප්‍රතිදානය දක්වන්න.

විකල්ප විසඳුම් හඳුනාගැනීම

ගැටලුවක් සඳහා විසඳුම් එකකට වඩා වැඩියෙන් පවතී නම් ඒවා විකල්ප විසඳුම් ලෙස හැඳින්වේ. මෙය ගැටලුවේ ස්වභාවය මත තීරණය වන්නකි.



නිදසුන

ඔබ පාසලට පැමිණෙන්නේ පාසල් බසයකින් යැයි සිතන්න. පාසලට පැමිණෙමින් සිටින විට බස් රථයේ දෝෂයක් නිසා අතරමග නැවතුණි නම් ඔබට පාසල වෙත පැමිණීමට හැකි වෙනත් ක්‍රම ගැන සිතනවා නේ ද? මෙසේ ඔබ සිතන්නේ පාසල වෙත යාමට හැකි වෙනත් විකල්ප විසඳුම් ය.

1. පාසලට එන වෙනත් පාසල් බස් රථයකින් පාසල වෙත පැමිණීම
2. ඔබ ළඟ මුදල් ඇති නම් ලංගම හෝ පෞද්ගලික බස් රථයකින් පාසල වෙත පැමිණීම
3. පාර දිගේ පයින් ම ගමන් කර පාසල වෙත පැමිණීම
4. කෙටි මාර්ගයකින් පයින් ගමන් කර පාසල වෙත පැමිණීම
5. දෙමවුපියන්ට මේ බව දැන්විය හැකි නම් ඔවුන්ගේ සහාය ඇති ව පාසල වෙත පැමිණීම
6. විශ්වාසවන්ත අයෙකුගේ සහාය ඇති ව මෝටර් රථයකින් හෝ මෝටර් සයිකලයකින් පාසල වෙත පැමිණීම

එදින අනිවාර්යයෙන් ම පාසලට පැමිණීමට අවශ්‍ය දිනයක් නම් මෙම විකල්ප විසඳුම් අතරින් හොඳ විසඳුමක් තෝරා ගන්නවා නේ ද?

ඒ අනුව යම් ගැටලුවකට විසඳුම් සමූහයක් (කුලකයක්) පවතී නම් ඒ පිළිබඳ ව සලකා බලා හොඳ විසඳුමක් තෝරා ගැනීම වඩාත් උචිත වේ.

මෙසේ ගැටලුවකට පවතින සියලු විසඳුම්, විසඳුම් අවකාශය (Solution Space) ලෙස හැඳින්වේ. පරිගණක ක්‍රම ලේඛනයේ දී ද විවිධ විසඳුම් හඳුනාගෙන හොඳ ම විසඳුම තෝරා ගැනීම කළ යුතු ය. එවිට සරල කෙටි ක්‍රමලේඛනයක් නිර්මාණය කිරීමට හැකි වේ.

නිදසුන 1

ඍජුකෝණාස්‍රයක පරිමිතිය සෙවීමට ඇති විසඳුම් අවකාශය සොයා බලමු.

මෙම ගැටලුවට අදාළ ආදාන, ප්‍රතිදාන සහ සැකසීම විශ්ලේෂණය කොට දක්වමු.

ආදාන : ඍජුකෝණාස්‍රයේ දිග හා පළල

සැකසීම : පරිමිතිය ගණනය කිරීම

ප්‍රතිදාන : පරිමිතිය

පරිමිතිය ගණනය කිරීම සඳහා විසඳුම් අවකාශය සොයමු.

1 වන විසඳුම. පරිමිතිය = දිග + පළල + දිග + පළල

2 වන විසඳුම. පරිමිතිය = (දිග × 2) + පළල × 2

3 වන විසඳුම. පරිමිතිය = (දිග + පළල) × 2

මෙම විසඳුම් අතරින් එකතු කිරීම පිළිබඳ ව පමණක් අවබෝධයක් ඇති අයෙකුට හොඳ ම විසඳුම ලෙස 1 වන විසඳුම තෝරා ගත හැකි ය. ගුණ කිරීම හා එකතු කිරීම පිළිබඳ ව අවබෝධයක් ඇති අයෙකුට හොඳ ම විසඳුම ලෙස 3 වන විසඳුම තෝරා ගත හැකි ය. එයට හේතුව එකතු කිරීමේ හා ගුණ කිරීමේ කාරක අවම ප්‍රමාණයක් පවතින බැවිනි.

නිදසුන් 2

ශිෂ්‍යයකු තොරතුරු හා සන්නිවේදන තාක්ෂණ විෂයය සඳහා ලබා ගත් ලකුණු සංඛ්‍යාව 35ට අඩු නම් ඔහු අසමත් බව හෝ 35 හෝ ඊට වැඩි නම් ඔහු සමත් බව ද දැක්වීම

ආදාන : ලකුණු සංඛ්‍යාව

සැකසීම : ලබා ඇති ලකුණු ප්‍රමාණය 35 සමග සැසඳීම

විසඳුම 1. ලකුණු 35ට අඩු නම්

ප්‍රතිඵලය = අසමත්

එසේ නොවේ නම්

ප්‍රතිඵලය = සමත්

විසඳුම 2. ලකුණු 35 හෝ ඊට වඩා වැඩි නම්

ප්‍රතිඵලය = සමත්

එසේ නොවේ නම්

ප්‍රතිඵලය = අසමත්

ප්‍රතිදාන : සමත් හෝ අසමත් බව

නිදසුන් 3

සංඛ්‍යා දෙකකින් විශාල සංඛ්‍යාව සෙවීම (පිටු අංක 03හි ගැටලුව 06 බලන්න.)

ආදානය කරනු ලබන සංඛ්‍යා දෙක n1 සහ n2 ලෙස ගනිමු.

විසඳුම 1. n1, n2 ට වඩා විශාල නම් විශාල සංඛ්‍යාව වන්නේ n1 වේ.

n2, n1 ට වඩා විශාල නම් විශාල සංඛ්‍යාව වන්නේ n2 වේ.

විසඳුම 2. n1න් n2 අඩු කරන්න.

එවිට ලැබෙන අගය 0ට වඩා විශාල නම් n1 විශාල සංඛ්‍යාව වේ.

එවිට ලැබෙන අගය 0ට වඩා කුඩා නම් n2 විශාල සංඛ්‍යාව වේ.

මෙසේ ගැටලුවලට පවතින විසඳුම් අතරින් හොඳ විසඳුම තෝරා ගැනීම ඉතා වැදගත් වේ.

1.2 ගැටලු විසඳීම සඳහා ඇල්ගොරිතම ගොඩනැගීම

ගැටලුවක් විසඳීම සඳහා අනුගමනය කරනු ලබන පියවර දැක්වීමේ ක්‍රමය ඇල්ගොරිතමයක් ලෙස හැඳින්වේ. මෙහි අවශ්‍යතාව වන්නේ පූර්ව සැලසුමක් සහිත ව නිවැරදි ව ගැටලුවකට විසඳුමක් සෙවිය හැකි ක්‍රියාමාර්ගයක් දැක්වීමයි.

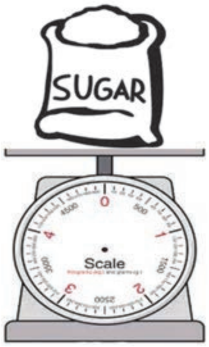
උදාහරණ 1 - ලිපියක් තැපැල් කිරීම

(1) ලිපිය ලිවීම
(2) ලිපිය නැවීම
(3) ලියුම් කවරයකට දැමීම
(4) ලිපිනය ලිවීම
(5) මුද්දර ඇලවීම
(6) තැපැල් කිරීම

ඉහත පියවර ලිපියක් තැපැල් කිරීම සඳහා වූ ඇල්ගොරිතමය වේ. මෙම ඇල්ගොරිතමයේ (1), (2) සහ (3) වන පියවර අනුපිළිවෙළින් ම සිදුකළ යුතු ය. (4) සහ (5) පියවර මාරුකළ හැකි ය. එයට හේතුව ලිපිනය ලියා මුද්දර ඇලවිය හැකි ය. එමෙන් ම මුද්දරය අලවා ලිපිනය ලිවිය හැකි ය.

ඒ අනුව ඇල්ගොරිතමයක අනුපිළිවෙළින් ම ක්‍රියාත්මක කළ යුතු පියවර පවතින අතර සමහර විට පියවර කිහිපයක අනුපිළිවෙළ වෙනස් කළ ද එමගින් ඉදිරිපත් කෙරෙන ක්‍රියාවලියට හානියක් සිදු නොවේ.

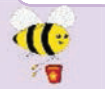
උදාහරණ 2 - තරාදියකින් සීනි ග්‍රෑම් 500ක් කිරා ගැනීම.



(1) සීනි කවරයකට දැමීම
(2) තරාදිය මත තබා පාඨාංකය කියවීම
(3) සීනි ග්‍රෑම් 500ට අඩු නම් පාඨාංකය ග්‍රෑම් 500ට සමාන වන තුරු කවරයට සීනි දැමීම
(4) සීනි ග්‍රෑම් 500ට වැඩි නම් පාඨාංකය ග්‍රෑම් 500ට සමාන වන තුරු කවරයෙන් සීනි ඉවත් කිරීම
(5) පාඨාංකය ග්‍රෑම් 500 වූ විට සීනි කවරය තරාදියෙන් ඉවතට ගැනීම

ඉහත පියවර මගින් සීනි ග්‍රෑම් 500 ක් කිරා ගැනීම සඳහා වූ ඇල්ගොරිතමය දැක්වේ.

ක්‍රියාකාරකම



ප්‍රාථමික පාසලක සිසු සිසුවියන් 183 ක් සිටිති. ඔවුන් ඕලු, නෙළුම් සහ මානෙල් ලෙස නිවාස තුනකට වෙන් කර නිවාසාන්තර ක්‍රීඩා උත්සවයක් පැවැත්වීමට විදුහල්පතිතුමා තීරණය කරයි. සිසු සිසුවියන් නිවාසවලට වෙන් කිරීම දැක්වෙන ඇල්ගොරිතමයක් ගොඩ නගන්න.

1.2.1 ඇල්ගොරිතම සංවර්ධනය කිරීම සඳහා පාලන ව්‍යුහ භාවිතය

ඇල්ගොරිතමයක ක්‍රියාකාරිත්වය අධ්‍යයනය කිරීමේ දී පාලන ව්‍යුහ (control structures) තුනක් හඳුනාගත හැකි වේ.

- අනුක්‍රමය (Sequence)

- වරණය (Selection)

- පුනර්කරණය (Iteration)

අනුක්‍රමය

ඇල්ගොරිතමයක පිහිටි පියවර කිහිපයක් හෝ සියල්ල ම හෝ ආරම්භක පියවරක සිට අවසාන පියවර දක්වා ඉහළ සිට පහළට එකෙල්ලේ අනුපිළිවෙළින් ක්‍රියාත්මක වීම අනුක්‍රමය ලෙස හැඳින්වේ.

උදාහරණ -

1. පඩිපෙළකින් ගමන් කිරීමේ දී පඩියෙන් පඩිය නැගීම හෝ බැසීම
2. පාසලේ 1 ශ්‍රේණියට ඇතුළත් වූ සිසුන් 11 ශ්‍රේණිය දක්වා අඛණ්ඩ ව අධ්‍යාපනය ලැබීම

ක්‍රියාකාරකම

අනුක්‍රමයක් සහිත සංසිද්ධින් තුනක් සඳහන් කරන්න.

වරණය

ඇල්ගොරිතමයක ඇතුළත් කොන්දේසියක් තෘප්ත වීම හෝ නොවීම හෝ අනුව ක්‍රියාත්මක කළ යුතු පියවර කුමක් ද යන්න තීරණය වන අවස්ථාවක් වරණයක් ලෙස හැඳින්වේ. මෙහි දී වරණ දෙකක් පවතින අතර කොන්දේසිය තෘප්ත වන්නේ නම් එක් වරණයක් ද තෘප්ත නොවන්නේ නම් අනෙක් වරණය ද තෝරා ගැනේ.

උදාහරණ -

1. පළමු ශ්‍රේණියට දරුවෙකු ඇතුළත් කිරීම:

   එම වසරේ ජනවාරි 31 දිනට දරුවකුගේ වයස අවුරුදු 5ට වැඩි නම්

   පාසලට ඇතුළත් කළ හැකි ය.

   එසේ නොවේ නම්

   පාසලට ඇතුළත් කළ නොහැකි ය.

2. විෂයකින් සමත් වීම:

   ලකුණු සංඛ්‍යාව 35 හෝ ඊට වැඩි නම්

   සමත් වේ

   එසේ නොවේ නම්

   අසමත් වේ.

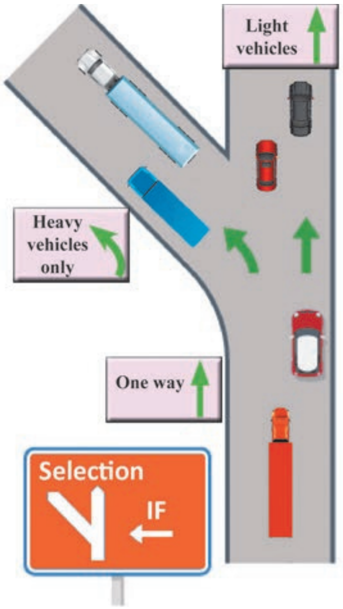
3. පොතක් මිල දී ගැනීම:

   පොතෙහි මිලට සමාන හෝ ඊට වැඩි හෝ මුදලක් ඔබට ඇත්නම්

   පොත මිල දී ගත හැකි ය.

   නැති නම්

   පොත මිල දී ගත නොහැකි ය.



ක්‍රියාකාරකම

1. වරණය සහිත සංසිද්ධි තුනක් සඳහන් කරන්න.
2. ශ්‍රී ලාංකික පුරවැසියකුට සිය සර්වජන ඡන්ද අයිතිය හිමි වන්නේ වයස අවුරුදු 18 සම්පූර්ණ වූ පසු ව නම් පහත දැක්වෙන හිස්තැන්වලට වඩාත් ගැළපෙන වචනය තෝරන්න.

   වයස අවුරුදු 18 හෝ ඊට .... (අඩු, වැඩි) නම්

   ඡන්දය ප්‍රකාශ කළ .... (හැකි ය, නොහැකි ය)

   නැතිනම්

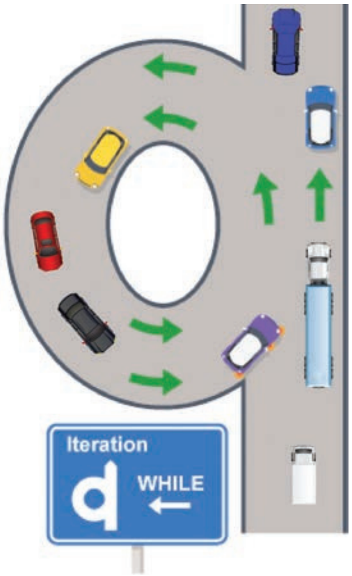
   ඡන්දය ප්‍රකාශ කළ .... (හැකි ය, නොහැකි ය)

පුනර්කරණය

ඇල්ගොරිතමයක ඇතුළත් පියවරක්, පියවර කිහිපයක් හෝ කොන්දේසියක් තෘප්ත වන තෙක් හෝ තෘප්ත ව පවතින තුරු හෝ නැවත නැවත සිදුවීම පුනර්කරණය ලෙස හැඳින්වේ.

උදාහරණ -

1. පන්තිභාර ගුරුතුමා හෝ ගුරුතුමිය ශිෂ්‍ය නාමලේඛනය ලකුණු කිරීමේ ක්‍රියාවලිය සලකමු.



(1) නාම ලේඛනයේ මුලින් ම ඇති නම කියවීම

(2) එම සිසුවා පැමිණ සිටී නම් 1 ලෙස සටහන් කිරීම

(3) එම සිසුවා පැමිණ නැති නම් 0 ලෙස සටහන් කිරීම

(4) ඊළඟ සිසුවාගේ නම අඩ ගැසීම

(5) නාම ලේඛනයේ ඇති නම් අවසන් වන තුරු ම පියවර 2 හෝ 3 සහ පියවර 4 සිදුකිරීම

2. ඡේදයක් කියවා එහි ඇති වචන සංඛ්‍යාව සෙවීමේ ක්‍රියාවලිය සලකමු.

(1) ඡේදයේ ආරම්භක වචනය කියවීම

(2) වචන සංඛ්‍යාව = 1

(3) ඊළඟ වචනය කියවීම

(4) වචන සංඛ්‍යාවට එකක් එකතු කිරීම

(5) ඡේදය අවසන් වන තුරු ම පියවර අංක 3 සහ 4 සිදු කිරීම

(6) ඡේදය කියවා අවසන් වූ විට වචන සංඛ්‍යාව ප්‍රකාශ කිරීම

ක්‍රියාකාරකම

1. පුනර්කරණය සහිත සිද්ධි 2ක් සඳහා පියවර වශයෙන් සඳහන් කරන්න.
2. 5 සිට 60 දක්වා ඇති 5හි ගුණාකාර ප්‍රකාශ කිරීම සඳහා වූ පුනර්කරණයට අදාළ හිස්තැන් පුරවන්න.

   පියවර 1   n = 5

   පියවර 2   n හි අගය ප්‍රකාශ කරන්න.

   පියවර 3   n හි අගයට 5 ක් එකතු කරන්න.

   පියවර 4   n හි අගය 60 වන තෙක් පියවර අංක ..... සහ ..... සිදු කරන්න.

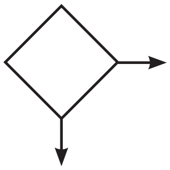
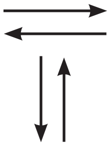
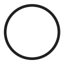
1.3 ඇල්ගොරිතම ඉදිරිපත් කිරීමට විවිධ මෙවලම් භාවිතය

ඇල්ගොරිතම නිර්මාණයේ දී සරල ව හා පහසුවෙන් ඉදිරිපත් කිරීම සඳහාත් ඇල්ගොරිතමය වඩාත් හොඳින් අවබෝධ කර ගැනීම සඳහාත් ගැලීම් සටහන් සහ ව්‍යාජ කේත යන මෙවලම් භාවිත කෙරේ.

1.3.1 ගැලීම් සටහන් (Flow charts)

ඇල්ගොරිතමය රූපමය ආකාරයෙන් පියවරෙන් පියවර ගොඩ නැගෙන හෝ ගොඩ නැගී ඇති ආකාරය හෝ ඉදිරිපත් කිරීම සඳහා ගැලීම් සටහන් යොදා ගැනේ. මෙහි දී එක් එක් ක්‍රියාව දැක්වීම සඳහා පහත දැක්වෙන සංකේත භාවිත කෙරේ. (වගුව 1.1)

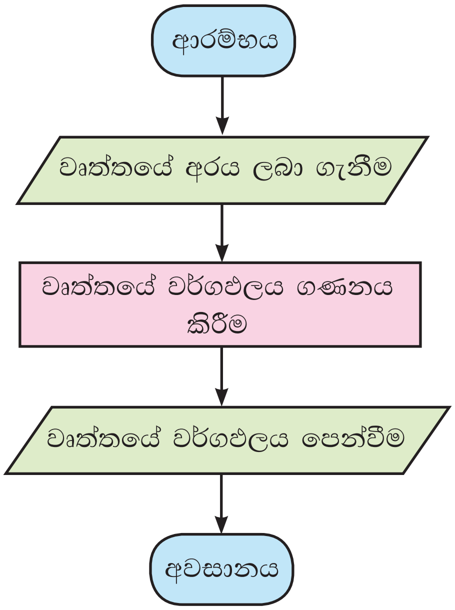
වගුව 1.1 - ගැලීම් සටහන් සංකේත

සංකේතය	අදහස
	ආරම්භය හෝ අවසානය
	ආදානය හෝ ප්‍රතිදානය
	ක්‍රියාවලිය
	තීරණය
	ගැලීම් දිශාව
	සම්බන්ධකය

අනුක්‍රමය සහිත ගැලීම් සටහන් :

ආරම්භක පියවරේ සිට අවසාන පියවර දක්වා ඇති පියවර සියල්ල ම එකින් එක අනුපිළිවෙළකට සිදුවීම අනුක්‍රමයක ලක්ෂණය වේ.

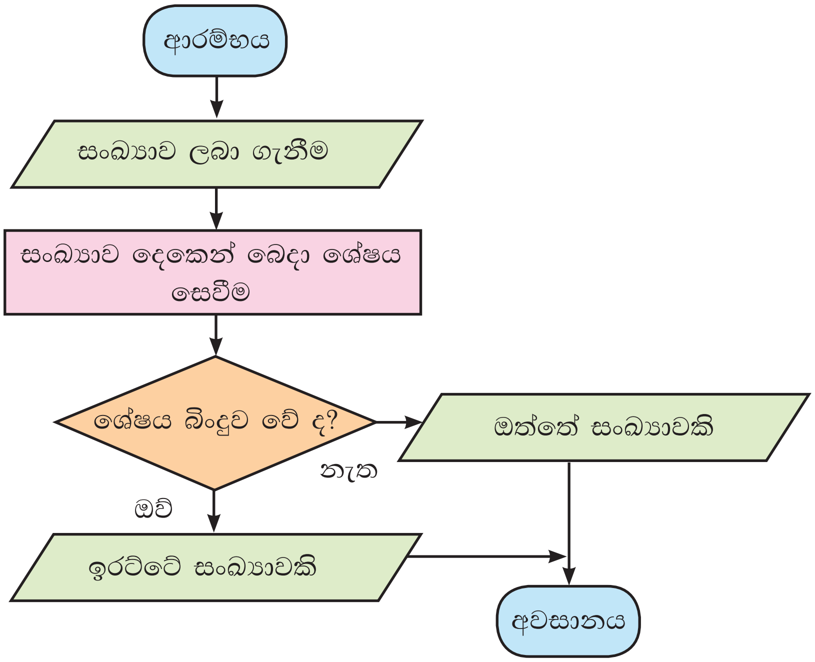
උදා - 1. වෘත්තයක වර්ගඵලය ගණනය කිරීම



වරණය සහිත ගැලීම් සටහන්

කොන්දේසියක් තෘප්ත වීම හෝ නොවීම හෝ අනුව ගැලීම් දිශාව තීරණය වීම වරණයක ඇති විශේෂත්වය යි.

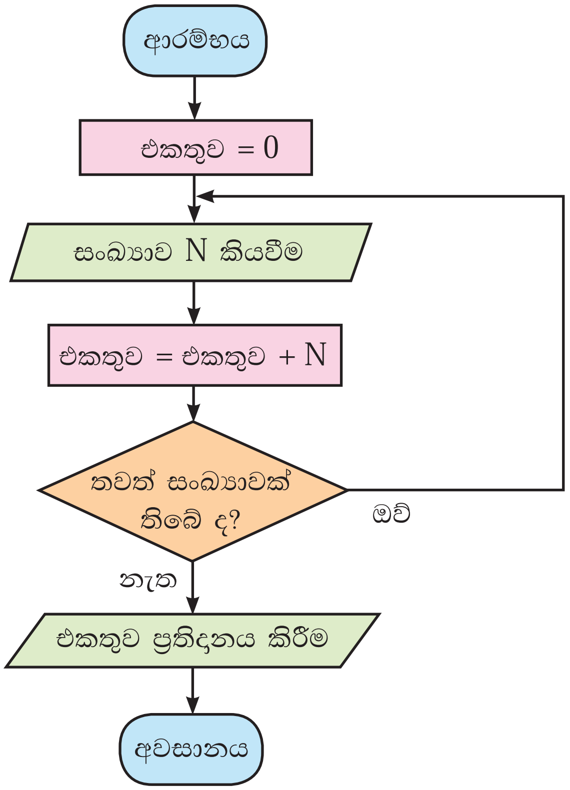
උදා - සංඛ්‍යාවක් ඔත්තේ ද, ඉරට්ටේ ද යන්න සෙවීම



පුනර්කරණ සහිත ගැලීම් සටහන් :

පියවර කිහිපයක්, කොන්දේසියක් තෘප්ත වන තෙක් හෝ තෘප්ත ව පවතින තාක් කල් හෝ නැවත නැවත සිදුවීම මෙහි ඇති විශේෂත්වයයි.

සංඛ්‍යා සමූහයක එකතුව සෙවීම



ක්‍රියාකාරකම

පහත දැක්වෙන ගැටලු සඳහා ගැලීම් සටහන් අඳින්න.

1. සෘජුකෝණාස්‍රයක පරිමිතිය සහ වර්ගඵලය සෙවීම
2. ආයතනයක සේවකයින්ගේ මූලික වැටුපට රුපියල් 5000.00ක දීමනාවක් ලබා දීමට තීරණය කර ඇත. ඒ අනුව නව වැටුප ගණනය කළ යුතු ව ඇත.
3. ලිපියක් තැපැල් කිරීමේ දී ගාස්තු අය කරන්නේ එහි බර ප්‍රමාණය අනුව ය. සම්මත බර ප්‍රමාණයට අඩු හෝ සමාන බර සහිත ලිපි සඳහා නියමිත ගාස්තු පමණක් ගෙවිය යුතු ය. සම්මත බර ප්‍රමාණයට වඩා වැඩි බර සහිත ලිපි සඳහා අමතර ගාස්තු ගෙවිය යුතු වේ.
4. 7 හි මුල් ම ගුණාකාර 12 දැක්වීම
5. පිටුව 8 - 9හි ඇති උදහරණ සඳහා ගැලීම් සටහන් අඳින්න.

1.3.2 ව්‍යාජ කේත (Pseudo codes)

ඇල්ගොරිතමයක් සරල ඉංග්‍රීසි වචන යොදා ගෙන ලිඛිත ව දැක්වීම ව්‍යාජ කේත මගින් සිදු වේ. මෙසේ ලියනු ලබන ව්‍යාජ කේත පරිගණක භාෂාවකින් ස්වායත්ත වේ. ව්‍යාජ කේත ඕනෑ ම පරිගණක භාෂා උපදෙස් බවට පරිවර්තනය කළ හැකි ය. එබැවින් ව්‍යාජ කේත ලිවීමෙන් පරිගණක ක්‍රමලේඛනය පහසු කරයි.

ඇල්ගොරිතමයේ එක් එක් ප්‍රකාශය සරල ඉංග්‍රීසි වචනවලින් දක්වන ආකාරය විමසා බලමු.

ආරම්භය	BEGIN
අවසානය	END
ආදානය	INPUT, READ, GET
ප්‍රතිදානය	OUTPUT, DISPLAY, SHOW
ක්‍රියාවලිය	PROCESS, CALCULATE
වරණය	IF … THEN .. .ELSE … ENDIF
පුනර්කරණය	FOR – DO
	WHILE – ENDWHILE
	REPEAT - UNTIL

ව්‍යාජ කේත ලිවීම

උදා 1 - වෘත්තයක වර්ගඵලය ගණනය කිරීම

```
BEGIN
    INPUT Radius
    CALCULATE Area = 22/7 × Radius × Radius
    DISPLAY Area
END.
```

උදා 2 - සංඛ්‍යාවක් ඔත්තේ ද, ඉරට්ටේ ද යන්න සෙවීම

```
BEGIN
    READ number as N
    CALCULATE  Remainder after number divided by 2
    IF Remainder = 0 THEN
        DISPLAY “Even number”
    ELSE
        DISPLAY “Odd number”
    ENDIF
END.
```

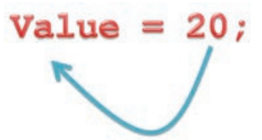
උදා 3 - සංඛ්‍යා සමූහයක එකතුව සෙවීම

```
BEGIN
    Total = 0
    REPEAT
        READ Number as N
        CALCULATE Total = Total + N
    UNTIL numbers are over
    DISPLAY Total
END.
```

උදා 4 - සංඛ්‍යා 10ක එකතුව සහ සාමාන්‍යය සෙවීම

```
BEGIN
    Total = 0
    Average = 0
    n = 1
    WHILE n <= 10

        READ number
        CALCULATE Total = Total + number
        n = n + 1
    ENDWHILE
    CALCULATE Average = Total/(n-1)
    DISPLAY Total, Average
END.
```

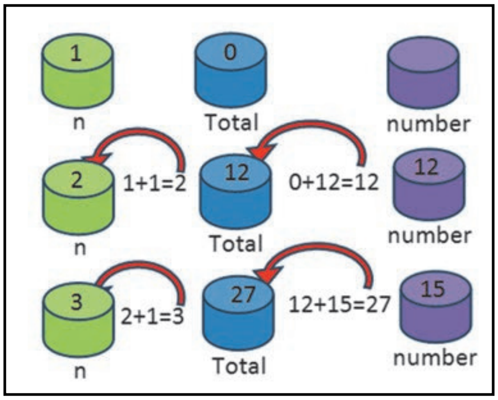


Value යන විචල්‍යයට 20 යන අගය පවරයි

ඉහත උදාහරණ 4 හි ව්‍යාජ කේතය පිළිබඳ ව දක්වා ඇති පහත කරුණු කෙරෙහි අවධානය යොමු කරන්න.

- Total, Average, number සහ n යනු විචල්‍ය (Variable) වේ.
    - Number යන විචල්‍ය සඳහා අගයන් ආදානය කරන විට ඊට සාපේක්ෂ ව Total, Average සහ n යන විචල්‍යවල අගයන් ද වෙනස් වේ.
- n විචල්‍ය මගින් පුනර්කරණය වන වාර ගණන නිරූපණය කරයි.
- Total = 0 සහ Average = 0 මගින් එම විචල්‍යවල ආරම්භක අගය 0 ලෙස පවරා ඇත.
    - මින් අදහස් කරන්නේ number විචල්‍ය සඳහා අගයන් ලබා ගැනීමට ප්‍රථමයෙන් Total සහ Average යන විචල්‍යවල සංඛ්‍යාත්මක අගයන් 0 ලෙස ගෙන ඇති බවයි.

- n = 1 මගින් n හි ආරම්භක අගය 1 ලෙස පවරා ඇත.
    - මින් අදහස් කරන්නේ n විචල්‍යයේ ආරම්භක අගය 1 වන බවයි.
- n <= 10 යනු පුනර්කරණය වීම පාලනය කරන කොන්දේසිය වේ.
- WHILE n <= 10 මගින් n හි අගය 10 වන තෙක් පුනර්කරණය විය යුතු බව දක්වයි.
    - n හි අගය 10 හෝ 10 ට අඩු අවස්ථාවල පුනර්කරණය සිදුවේ. එනම් n <= 10 යන කොන්දේසිය සත්‍ය ව පවතින තුරු පුනර්කරණය සිදුවේ. nහි අගය 11 වන විට පුනර්කරණය වීම නවතී. එවිට කොන්දේසිය අසත්‍ය වේ.



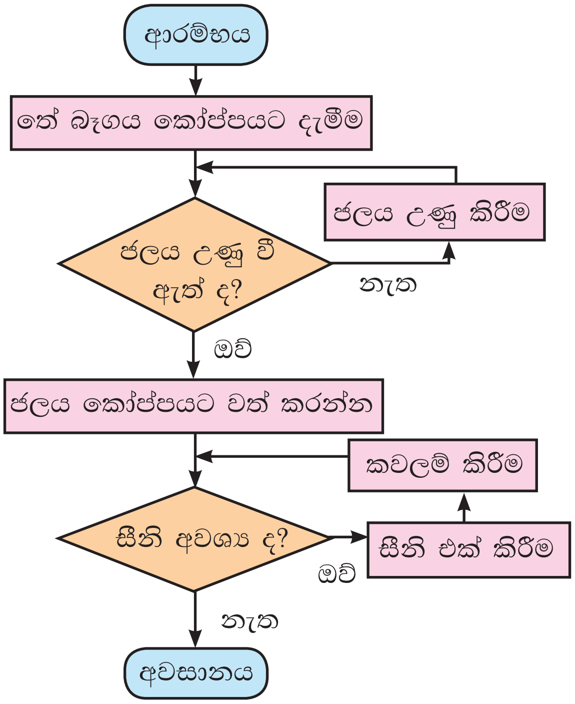
- READ මගින් number විචල්‍යය සඳහා අගයක් ලබා ගැනීම දක්වයි.
- Total + Total = number මගින් number සඳහා ලබා ගත් අගය Total විචල්‍යයට එකතු කර එම අගය නැවත Total විචල්‍යයට ම පැවරීම දක්වයි.
- n = n + 1 මගින් පුනර්කරණය වන වාර ගණන ගණනය වේ. මෙහි දී number විචල්‍යය සඳහා අගයක් ලබා ගත් පසු n හි අගයට 1 ක් එකතු කර එම අගය නැවත n විචල්‍යයට ම පවරනු ලබයි.
- ENDWHILE මගින් පුනර්කරණය අවසන් කරන සීමාව දක්වයි. ඒ අනුව READ number, Total = Total + number සහ n = n + 1 යන ප්‍රකාශන පමණක් n <= 10 යන කොන්දේසිය තෘප්ත ව පවතින තෙක් පුනර්කරණය වේ.
- පුනර්කරණය වීම නවතින විට n හි අගය 11 වේ. එම අවස්ථාවේ කොන්දේසිය අසත්‍ය වේ.
- Average = Total/(n-1) මගින් Total හි අවසන් අගය n-1 න් බෙදා ලැබෙන අගය Average විචල්‍යයට පවරනු ලබයි. එනම් Average විචල්‍යය මගින් සාමාන්‍යය ගණනය කෙරේ.
- DISPLAY Total, Average මගින් ආදානය කරන ලද සංඛ්‍යා දහයේ එකතුව සහ සාමාන්‍යය ප්‍රතිදානය කරයි.

නිරීක්ෂණය

- විචල්‍යයක් සඳහා අගයන් පැවරෙන විට පෙර පැවති අගය නැති වී යයි.
- Total = Total + number යන ප්‍රකාශනය ක්‍රියාත්මක වන විට number විචල්‍යය සඳහා ඇති අගය Total විචල්‍යයේ පැවති අගය සමග එකතු වීමෙන් පසු ලැබෙන අගය Total විචල්‍යය වෙත ම පැවරේ.
- Total = Total + number යනු ගණිතමය සමීකරණයක් නොවේ.

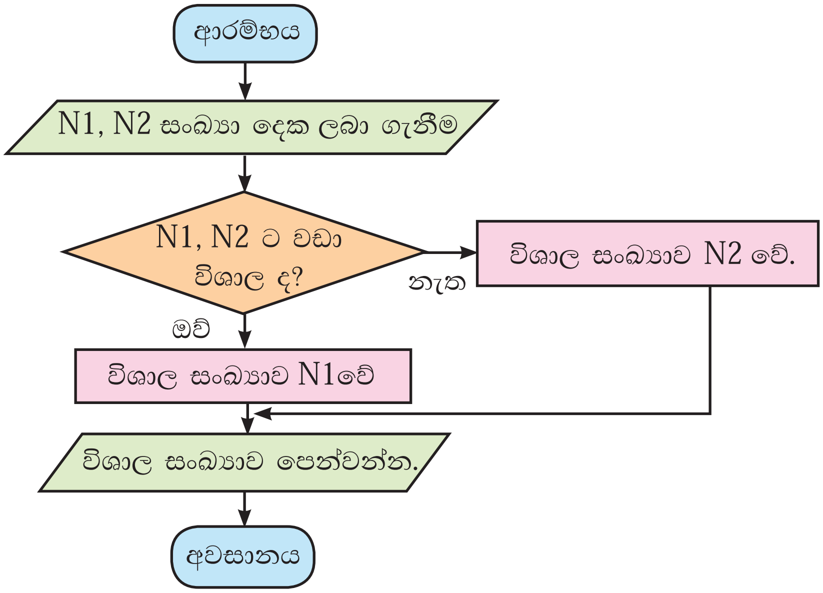
1.3.3 ගැලීම් සටහන් ව්‍යාජ කේතවලට පරිවර්තනය කිරීම

එක ම ඇල්ගොරිතමයක් ගැලීම් සටහනකින් මෙන් ම ව්‍යාජ කේතයකින් ද නිරූපණය කළ හැකි ය. එබැවින් ගැලීම් සටහනක් ව්‍යාජ කේතයක් බවට පරිවර්තනය කරන ආකාරය විමසා බලමු.



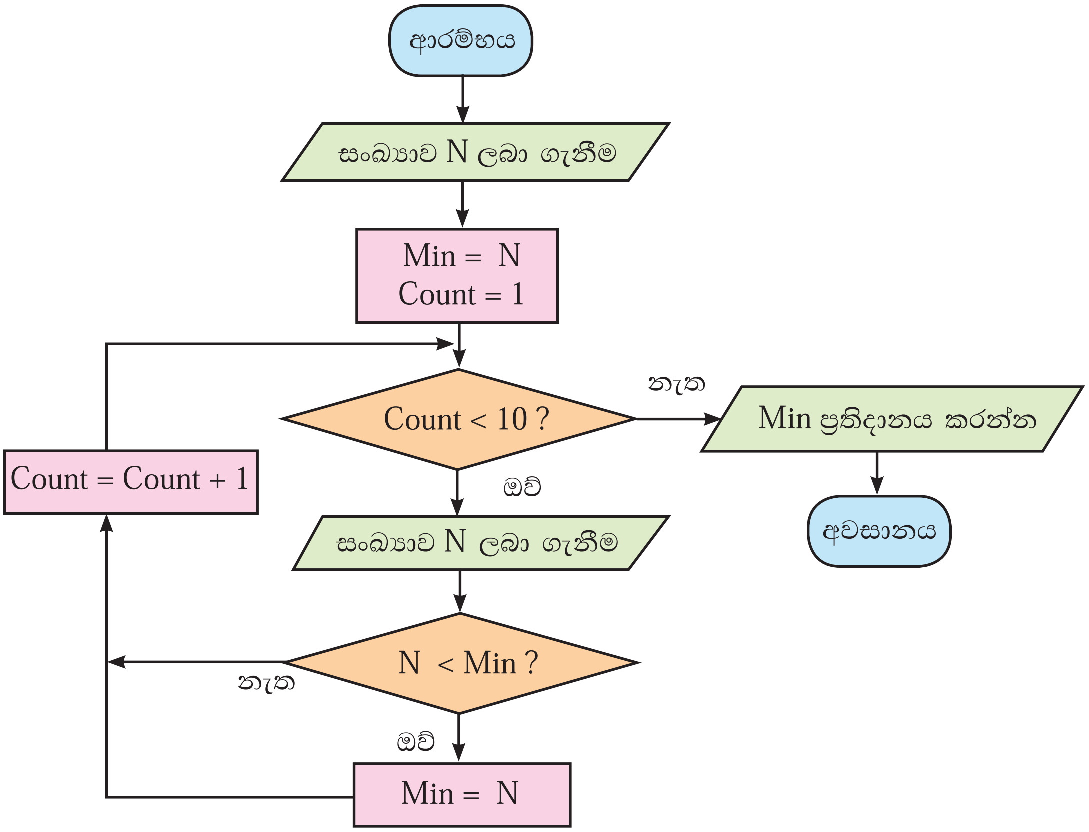

```
BEGIN
    Put tea bag in cup
    WHILE (not Water boiled)
        Boil water
    ENDWHILE
    Pour water in cup
    WHILE (sugar needed)
        Add sugar
        Stir tea
    ENDWHILE
END
```

උදා 1 - අසමාන සංඛ්‍යා දෙකක් අතරින් විශාල සංඛ්‍යාව සෙවීම



```
BEGIN
    READ N1, N2
    IF N1 > N2 THEN
        Large = N1
    ELSE
        Large = N2
    ENDIF
    DISPLAY Large
END
```

උදා 2 - සංඛ්‍යා දහයක් අතරින් කුඩා ම සංඛ්‍යාව සෙවීම.

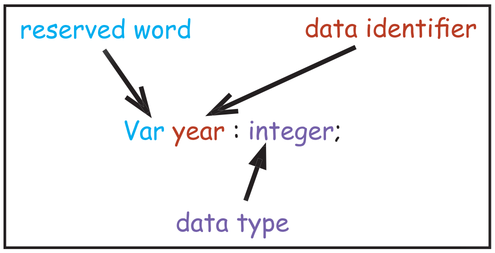


```
BEGIN
        INPUT Number as N
        Min = N
        Count = 1
        WHILE Count<10
                INPUT Number as N
                IF N < Min  Then
                        Min = N
                ENDIF
                Count = Count + 1
        ENDWHILE
        PRINT  Min
END.
```

1.4 පැස්කල් ක්‍රමලේඛන භාෂාව

1.4.1 හඳුන්වන (Identifiers)

හඳුන්වනයක් යනු විචල්‍යයක් (Variable), නියතයක් (Constant) හෝ ක්‍රමලේඛනයක් (Program) හැඳින්වීම සඳහා යොදා ගන්නා නමකි. මෙම හඳුන්වනයක් යෙදීමේ දී පහත දැක්වෙන සාමාන්‍ය නීති-රීති අනුගමනය කළ යුතු ය.



- පැස්කල් භාෂාව හඳුන්වා දීම සඳහා භාවිත කරනු ලබන ඇවුරුණු පද (Reserved words) යොදා නොගත යුතු ය. ඕනෑ ම භාෂාවක ඇවුරුණු පද හඳුන්වන සඳහා යොද ගත නොහැක.

  උදා - BEGIN, END ආදිය වලංගු නැත.
  (eg : A-Z, a-z)

- ඉංග්‍රීසි හෝඩියේ අක්ෂරයකින් ආරම්භ කළ යුතු ය.
- හඳුන්වනයේ පළමු අක්ෂරයට පසු ව අක්ෂර (a-z, A-Z) හෝ සංඛ්‍යාංක (0-9) සහ යටිඉර (_) යොදා ගත හැකි ය.

  උදා - Student_name

- පැස්කල් භාෂාවේ දී හඳුන්වන සඳහා අක්ෂර බේදය බල නොපායි.

  (උදා - Art, art, ART යනු එක ම හඳුන්වනයකි.)
- වචන අතර පරතරයක් (space) නොපැවතිය යුතු ය.

  උදා - Student Name - වලංගු නැත.
- හඳුන්වනයක් තුළ පහත අනුලක්ෂණ අඩංගු නොවිය යුතු ය.

  ~ ! @ # $ % ^ & * ( ) - + = { } [ ] : ; ' " <> ? , . / | \

  නමුත් underscore (_) ලකුණ පමණක් වලංගු වේ.
- හඳුන්වන සඳහා අර්ථවත් නාම යොදා ගැනීමෙන් ක්‍රමලේඛ තේරුම් ගැනීමට පහසු වේ.

වලංගු හඳුන්වන සඳහා උදාහරණ -

Sum, SUM, Total_Nos, Num1, FirstName, Last_Name

වලංගු නොවන හඳුන්වන සඳහා උදාහරණ -

$75, Average Marks, 9A, Last-name

1.4.2 ඇවිරුණු පද (Reserved words)

පැස්කල් භාෂාවේ භාවිත කරන ඇවුරුණු පද පැස්කල් භාෂාව හඳුන්වා දීම සඳහා යොදා ගෙන ඇත. එබැවින් හඳුන්වන (Idntifier) සඳහා ඇවුරුණු පද භාවිත නො කෙරේ.

ඇවුරුණු පද ක්‍රමලේඛන භාෂාවෙන් භාෂාවට වෙනස් වේ. පැස්කල්වල භාවිත කරන ඇවුරුණු පද පහත දැක්වේ.

and	exports	mod	shr
asm	file	nil	string
array	for	not	then
begin	function	object	to
case	goto	of	type
const	if	or	unit
constructor	implementation	packed	until
destructor	in	procedure	uses
div	inherited	program	var
do	inline	record	while
downto	interface	repeat	with
else	label	set	xor
end	library	shl	

1.4.3 පැස්කල් භාෂාවේ සම්මත දත්ත ප්‍රරූප (Data type)

ක්‍රමලේඛය ක්‍රියාත්මක වන විට ආදාන සහ ක්‍රියාවලිය තුළින් ගණනය වූ ප්‍රතිඵල පරිගණක මතකය තුළ රඳවා තබා ගැනීමට අවශ්‍ය වේ. ඒ සඳහා අවශ්‍ය ඉඩ ප්‍රමාණය තීරණය කරනු ලබන්නේ දත්ත ප්‍රරූපය මත ය. එබැවින් දත්ත ප්‍රරූප පිළිබඳ ව ක්‍රමලේඛකයකු දැන සිටීම ඉතා වැදගත් වේ.

දත්ත ප්‍රරූප හා ඒවායේ පරාස පහත දැක්වේ.

Integer - ධන හෝ ඍණ නිඛිල සංඛ්‍යා/පූර්ණ සංඛ්‍යා

උදා - 0, 46, -12

Real - ධන හෝ ඍණ තාත්වික සංඛ්‍යා/දශම සංඛ්‍යා

උදා - 0.0, 25.68

Boolean

True හෝ False

Char - යතුරු පුවරුවේ ඇති ඕනෑ ම අනුලක්ෂණයක්

උදා -‘k’, ‘#’, ‘7’

String - ඕනෑ ම අනුලක්ෂණ අනුක්‍රමයක්

උදා -‘ICT’, ‘programming’, 'Sri Lanka'

VARIABLE NAME	VALUE	TYPE
number	- 123	integer
sum	456	integer
character	‘B’	char
book	‘Mathematics’	string

Char සහ String යන දත්ත ප්‍රරූප තනි උඩු කොමා තුළ (‘ ’) (single quotation) දක්වනු ලැබේ.

1.4.4 ක්‍රමලේඛනයේ දී යොදා ගන්නා විචල්‍ය (variable) සහ නියත (constant)

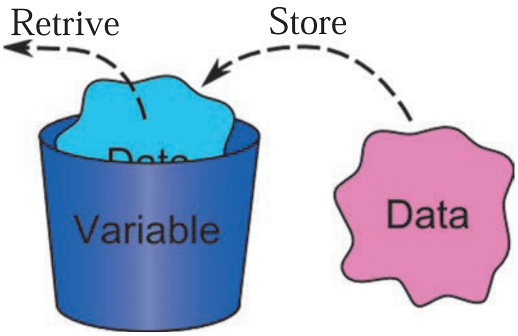
විචල්‍ය (Variable)

ක්‍රමලේඛය ක්‍රියාත්මක වන විට හඳුන්වනය සඳහා පවරන ලද අගයන් වෙනස් වන හඳුන්වන, විචල්‍යයක් ලෙස නම් කෙරේ.

පැස්කල් භාෂාවේ දී විචල්‍ය අර්ථ දැක්වීම සඳහා "var" යන ඇවුරුණු පදය යොදා ගැනේ.

උදා -

```
Var  count : integer;
Var  a,b : Real;
Var     n1, n2 : integer;
        Avg : real ;
        Pass : boolean;
        Character : char;
        Name,school : String;
```



විචල්‍යයක් සඳහා නමක් පවතින අතර එහි නිශ්චිත දත්ත ප්‍රරූපයකට අදළ දත්ත තැන්පත් කෙරේ.

A variable has a <u>name</u>, stores a <u>value</u> of the <u>declared</u> type.

නියත (Const)

ක්‍රමලේඛය ක්‍රියාත්මක වන විට හඳුන්වනය සඳහා පවරන ලද අගයන් වෙනස් නොවන හඳුන්වන, නියත ලෙස නම් කෙරේ. පැස්කල් භාෂාවේ දී නියත අර්ථ දැක්වීම සඳහා "const" යන ඇවුරුණු පදය යොදා ගැනේ.

උදා -

```
Const        max = 100;
Const        pi = 22/7;
```

නිරීක්ෂණය

- ක්‍රමලේඛය ධාවනය වීමේ දී විචල්‍යයක් සඳහා විවිධ අගයන් පවතින අතර නියතයක අගය නොවෙනස් ව පවතී.

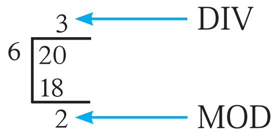
1.5 කාරක (Operators)

කාරක නොමැති ව ගණනය කිරීම, සංසන්දනය කිරීම සහ තාර්කික ප්‍රකාශන ගොඩ නැගීම කළ නොහැකි ය. එබැවින් ක්‍රමලේඛ ලිවීමට කාරක අත්‍යවශ්‍ය වේ.

මූලික කාරක වර්ග කිහිපයකි.

1. අංක ගණිතමය කාරක

කාරකය	භාවිතය	ප්‍රකාශය	ප්‍රතිඵලය
+	එකතු කිරීම	6 + 3	9
-	අඩු කිරීම	7 - 5	2
*	ගුණ කිරීම	2 * 5	10
/	බෙදීම	10/4	2.50
DIV	පූර්ණ සංඛ්‍යාමය බෙදීම	20 DIV 6	3
MOD	බෙදීමෙන් පසු ශේෂය	20 MOD 6	2



2. සැසඳුම් කාරක

අගයන් හෝ ප්‍රකාශන සංසන්දනය කිරීම සඳහා සැසඳුම් කාරක යොදා ගැනේ. සැසඳුම් කාරකයක් අඩංගු ප්‍රකාශනයක අවසන් ප්‍රතිඵලය සෑම විට ම බූලියානු අගයක් ගනී. එනම් ප්‍රකාශනය සත්‍ය හෝ අසත්‍ය වේ.

කාරකය	භාවිතය	ප්‍රකාශය	ප්‍රතිඵලය
>	විශාල	7 > 3	සත්‍ය
>=	විශාල හෝ සමාන	8 >= 8	සත්‍ය
<	කුඩා	3 < 2	අසත්‍ය
<=	කුඩා හෝ සමාන	4 <= 6	සත්‍ය
=	සමාන	3 = 1	අසත්‍ය
< >	අසමාන	2 < > 5	සත්‍ය

3. තාර්කික කාරක

ප්‍රකාශන දෙකක් හෝ කිහිපයක් ඒකාබද්ධ කිරීම සඳහා තාර්කික කාරක (Logical Operator) යොදා ගැනේ. මෙහි වැඩිදුර අධ්‍යයනය සඳහා 10 ශ්‍රේණියේ දී ඔබ හැදෑරූ මූලික තාර්කික ද්වාර බලන්න.

i) AND කාරකය

"(පළමු ප්‍රකාශනය) AND (දෙවන ප්‍රකාශනය)" ලෙස මූලික ව යොදනු ලබයි. එහි දී පළමු හා දෙවන ප්‍රකාශන සත්‍ය හෝ අසත්‍ය වීම අනුව AND කාරකය ගැනීමෙන් ලැබෙන ප්‍රතිඵලය ද සත්‍ය හෝ අසත්‍ය හෝ වේ. පහත වගුව මගින් AND කාරකයේ ක්‍රියාකාරිත්වය දැක්වේ.

පළමු ප්‍රකාශනය	දෙවන ප්‍රකාශනය	(පළමු ප්‍රකාශනය) AND (දෙවන ප්‍රකාශනය)
අසත්‍ය	අසත්‍ය	අසත්‍ය
අසත්‍ය	සත්‍ය	අසත්‍ය
සත්‍ය	අසත්‍ය	අසත්‍ය
සත්‍ය	සත්‍ය	සත්‍ය

උදා -
1. (වර්ෂාපතනය > 56) AND (උෂ්ණත්වය < 30),
2. (උස > 60) AND (වයස < 15)

3. (3 >= 2) AND (3 < > 3) යන ප්‍රකාශනය සලකමු.
   3 >= 2 යන ප්‍රකාශනය සත්‍ය වේ. 3 < > 3 යන ප්‍රකාශනය අසත්‍ය ද වේ.
   එබැවින් සමස්ත ප්‍රකාශනය ද අසත්‍ය වේ.

වැදගත්

- අවම වශයෙන් එක් ප්‍රකාශනයක් හෝ අසත්‍ය වූ විට AND කාරකය සහිත සමස්ත ප්‍රකාශනය ද අසත්‍ය වේ.
- සියලු ම ප්‍රකාශන සත්‍ය වන විට පමණක් AND කාරකය සහිත ප්‍රකාශනය සත්‍ය වේ.

ii) OR කාරකය

"(පළමු ප්‍රකාශනය) OR (දෙවන ප්‍රකාශනය)" ලෙස මූලික ව යොදනු ලබයි. එහි දී පළමු හා දෙවන ප්‍රකාශන සත්‍ය හෝ අසත්‍ය හෝ වීම අනුව OR කාරකය ගැනීමෙන් ලැබෙන ප්‍රතිඵලය ද සත්‍ය හෝ අසත්‍ය හෝ වේ. පහත වගුව මගින් OR කාරකයේ ක්‍රියාකාරිත්වය දැක්වේ.

පළමු ප්‍රකාශනය	දෙවන ප්‍රකාශනය	(පළමු ප්‍රකාශනය) OR (දෙවන ප්‍රකාශනය)
අසත්‍ය	අසත්‍ය	අසත්‍ය
අසත්‍ය	සත්‍ය	සත්‍ය
සත්‍ය	අසත්‍ය	සත්‍ය
සත්‍ය	සත්‍ය	සත්‍ය

උදා -
1. (උෂ්ණත්වය > 30) OR (වර්ෂාපතනය < 55)
2. (3 >= 2) OR (3 < > 3) යන ප්‍රකාශනය සලකමු.
   3 >= 2 යන ප්‍රකාශනය සත්‍ය වේ. 3 < > 3 යන ප්‍රකාශනය අසත්‍ය වේ.
   එබැවින් සමස්ත ප්‍රකාශනය සත්‍ය වේ.

වැදගත්

- අවම වශයෙන් එක් ප්‍රකාශනයක් හෝ සත්‍ය වූ විට OR කාරකය සහිත ප්‍රකාශනය ද සත්‍ය වේ.
- සියලු ම ප්‍රකාශන අසත්‍ය වන විට පමණක් OR කාරකය සහිත ප්‍රකාශනය අසත්‍ය වේ.

iii) NOT කාරකය

සෑම විට ම සත්‍ය ප්‍රකාශනයක් NOT කාරකය මගින් අසත්‍ය ලෙස දක්වන අතර අසත්‍ය ප්‍රකාශනයක් NOT කාරකය මගින් සත්‍ය ලෙස දක්වයි.

ප්‍රකාශනය	NOT (ප්‍රකාශනය)
අසත්‍ය	සත්‍ය
සත්‍ය	අසත්‍ය

උදා -
1. NOT (උෂ්ණත්වය > 30)
2. NOT (5 = 5) අසත්‍ය ප්‍රකාශනයකි.
   5 = 5 යන ප්‍රකාශනය සත්‍ය වේ.
   එබැවින් NOT(5 = 5) යන ප්‍රකාශනය අසත්‍ය ලෙස දක්වයි.

කාරක ප්‍රමුඛතාව

පැස්කල් ප්‍රකාශන ක්‍රියාත්මක වීමේ දී පහත දැක්වෙන කාරක ප්‍රමුඛතා අනුපිළිවෙළ (Operator Precedence) අනුගමනය කෙරේ.

ප්‍රමුඛතා මට්ටම	කාරක	වැඩි ය
1	NOT	↑
2	* / DIV MOD AND	↑
3	+ - OR	↑
4	= < > < <= > >=	අඩු ය

පැස්කල් ප්‍රකාශන ඇගයීම කරන අයුරු

උදා - (1)

5 + 14 MOD 4
5 + 2
7

උදා - (2)

3 + 7 DIV 2
3 + 3
6

උදා - (3)

16 / 4 * 2
4 * 2
8

උදා - (4)

NOT (8 MOD 2 > 5)
NOT (0 > 5)
NOT(False)
True

උදා - (5)

4 >= 4 AND NOT(7 > 9)
True AND NOT(False)
True AND True
True

සාමාන්‍ය පැස්කල් ක්‍රමලේඛයක අඩංගු මූලික කොටස් හඳුනා ගනිමු.

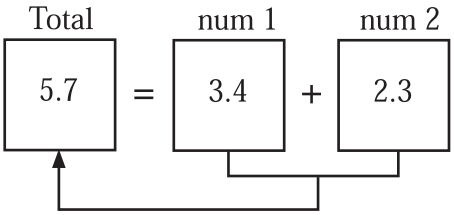
```
program addNum (input,output);          ← ක්‍රමලේඛයේ නම
var
        num1,num2,total : integer;  }
        avg: real;                  }   විචල්‍ය හඳුන්වා දීම

Begin                                   ← ප්‍රධාන ක්‍රමලේඛයේ ආරම්භය
        Write('Welcome Pascal Programming');           ← ප්‍රතිදාන
        {calculate total and  average of two numbers}   ← විවරණ (comments) දැක්වීම
        Writeln('Enter First Number');                  ← ප්‍රතිදාන
        Read(num1);                                     ← ආදාන
        Writeln('Enter Second Number');                 ← ප්‍රතිදාන
        Read(num2);                                     ← ආදාන
        total := num1 + num2;  }
        avg := total/2;        }  ක්‍රියාවලිය දැක්වෙන ප්‍රකාශ
        writeln('Total is ' , total);                   ← ප්‍රතිදාන
        writeln('Average is ' , avg);                   ← ප්‍රතිදාන
End.                                    ← ප්‍රධාන ක්‍රමලේඛයේ අවසානය
```

සටහන - විවරණ දැක්වීම සඳහා (*..........*) ද භාවිත කළ හැකි ය.

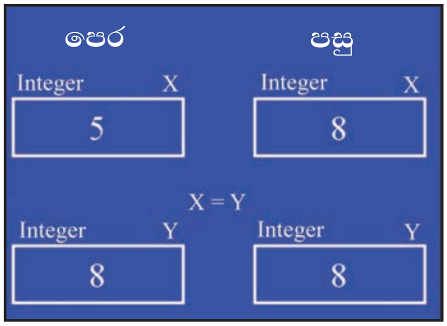
- මෙහි "program", "input" සහ "output" යනු ඇවිරුණු පද වේ.
- "addNum" යනු හඳුන්වනයකි. මෙය ක්‍රමලේඛයේ නාමය යි. ක්‍රමලේඛයේ නාමය සමග වරහන් තුළ input, output ලෙස දැක්වීම අත්‍යවශ්‍ය නො වේ.
- ආදානය සඳහා read() සහ readln() ඇවිරුණු පද භාවිත කෙරේ.
  - Read(num1); ප්‍රකාශනය මගින් num1 වූ විචල්‍ය සඳහා දත්ත ආදානය කර ගැනේ.
  - readln() මගින් දත්ත ආදානය කර ගන්නේ නව පේළියක සිට ය.

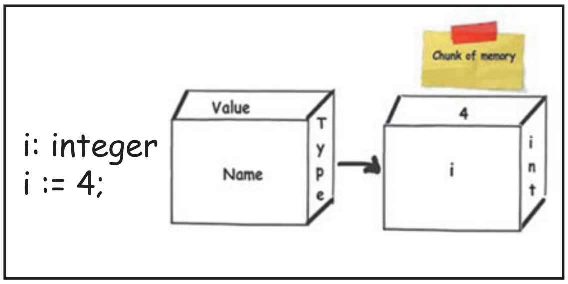
- ප්‍රතිදානය සඳහා write() සහ writeln() ඇවිරුණු පද භාවිත කෙරේ.
    - Write (‘Welcome Pascal Programming’); ප්‍රකාශනය මගින් Welcome Pascal Programming නම් වූ පාඨ කොටස ප්‍රතිදානය කරයි.
    - writeln (‘Average is ’, avg); ප්‍රකාශනය මගින් නව පේළියක Average is යන පාඨ කොටස සමග avg විචල්‍යයේ අගය ප්‍රතිදානය කරයි.



Pascal ප්‍රකාශන (statements) ලිවීමේ දී;

- ප්‍රකාශනයක කෙළවරේ semi-colon (;) යොදනු ලබයි. semi-colon මගින් ප්‍රකාශනය අවසන් බව කියවේ.
- total := num1 + num2; යන ප්‍රකාශනයෙන් සිදුවන්නේ num1 සහ num2 විචල්‍ය දෙකේ අගයන් එකතු කර total නම් වූ විචල්‍යයට පවරනු ලැබීමයි.
- මෙහි දී භාවිත කරන ":=" පැවරුම් කාරකය (Assignment Operator) ලෙස හැඳින්වේ.

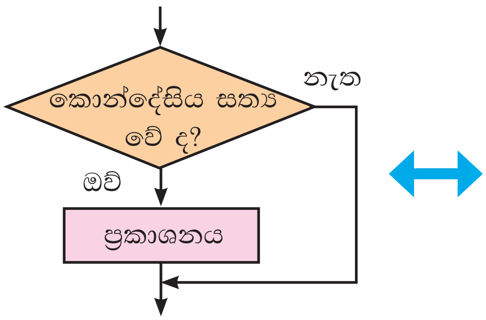




1.6 වරණ පාලන ව්‍යුහය සම්බන්ධ වන ක්‍රමලේඛ නිර්මාණය

සරල IF ප්‍රකාශය

වරණ පාලන ව්‍යුහය සරල ම ආකාරයෙන් පහත දැක්වේ. මෙහි ආකාර 02 කි.



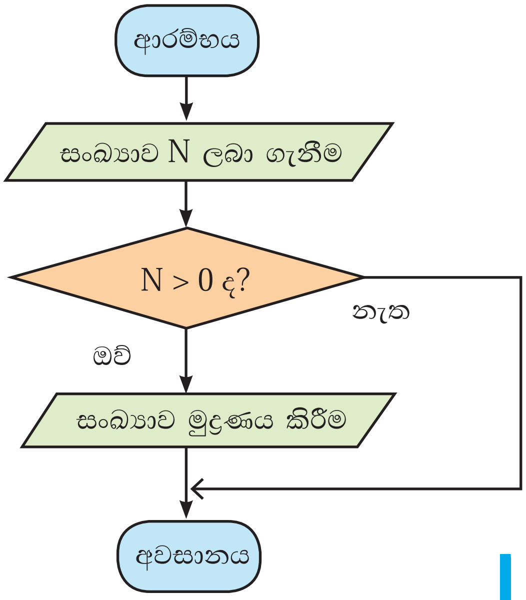

```
IF Condition THEN
        Statement
ENDIF
```

ව්‍යාජ කේතය

i) IF... THEN.... ENDIF

මෙහි දී කොන්දේසිය තෘප්ත වන්නේ නම් පමණක් ප්‍රකාශනය ක්‍රියාත්මක වේ.

උදා 1 - ආදානය කරන සංඛ්‍යාව ධන නම් පමණක් එම සංඛ්‍යාව මුද්‍රණය කිරීම.



ගැලීම් සටහන

```
Begin
  Input N
  IF  N > 0
THEN
        Print N
  ENDIF
End.
```

ව්‍යාජ කේතය

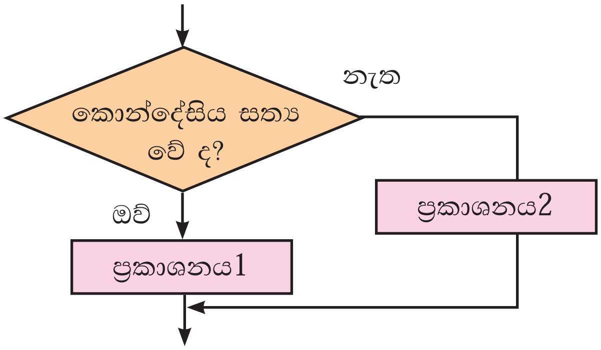
```
Program positiveNo(input, output);
Var  N : integer;
Begin
        Writeln('Enter Number');
        Read(N);
        If N > 0 then
        Writeln('Positive Number');
End.
```

පැස්කල් ක්‍රමලේඛය

ii) IF... THEN.... ELSE .... ENDIF

මෙහි දී කොන්දේසිය තෘප්ත වන්නේ නම් ප්‍රකාශනය 1 ද තෘප්ත නොවේ නම් ප්‍රකාශනය 2 ද ක්‍රියාත්මක වේ.

උදා 2 - අසමාන සංඛ්‍යා දෙකක් අතරින් විශාල සංඛ්‍යාව සෙවීම



ගැලීම් සටහන

```
IF  Condition  THEN
        Statement1
ELSE
        Statement2
ENDIF
```

ව්‍යාජ කේතය

```
program LargeNo(input,output);
Var  N1,N2,Large: integer;
Begin
        Writeln('Enter Two Numbers');
        Read(N1,N2);
        If N1 > N2 then
                Large := N1
        Else
                Large := N2;
        Writeln('Large Number is ' , Large);
End.
```

පැස්කල් ක්‍රමලේඛය

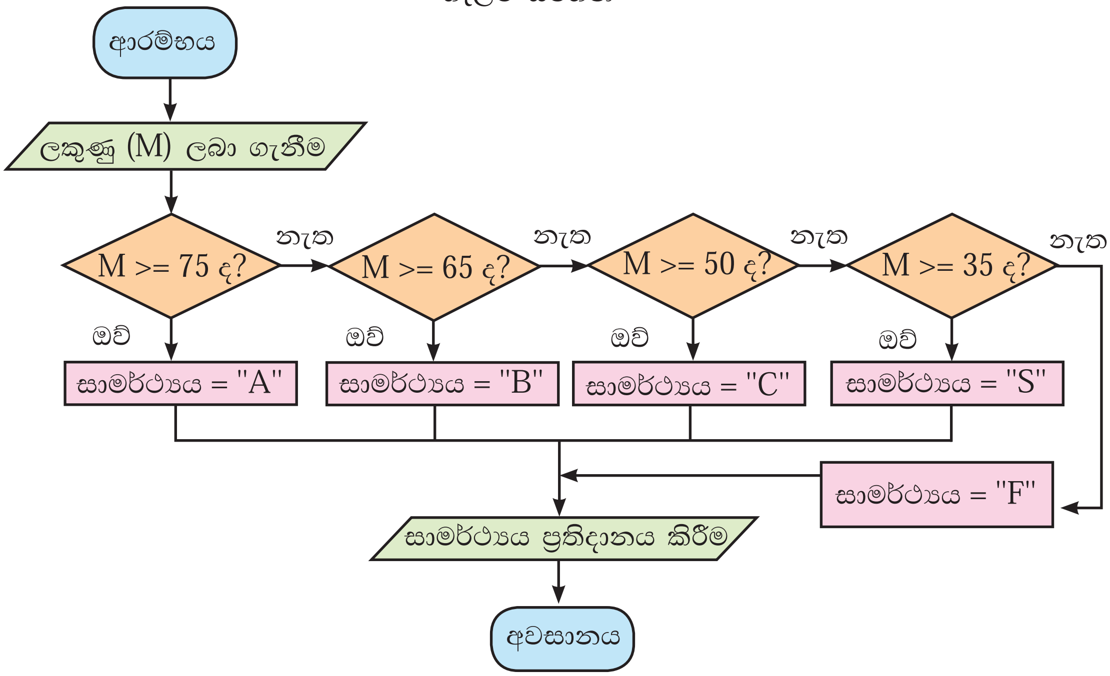
NESTED IF

කොන්දේසියකට පසු ව කොන්දේසියක් වශයෙන් කොන්දේසි සමූහයක් යෙදෙන අවස්ථාවක NESTED IF යොදා ගැනේ.

i) කේවල විචල්‍යයකට බහු කොන්දේසි ඇති විට NESTED IF භාවිතය

උදා 3 - විෂයකට සිසුවකු ලබා ගත් ලකුණු ආදානය කළ විට ඊට අදාළ සාමර්ථ්‍යය සෙවීම

ගැලීම් සටහන



ව්‍යාජ කේතය

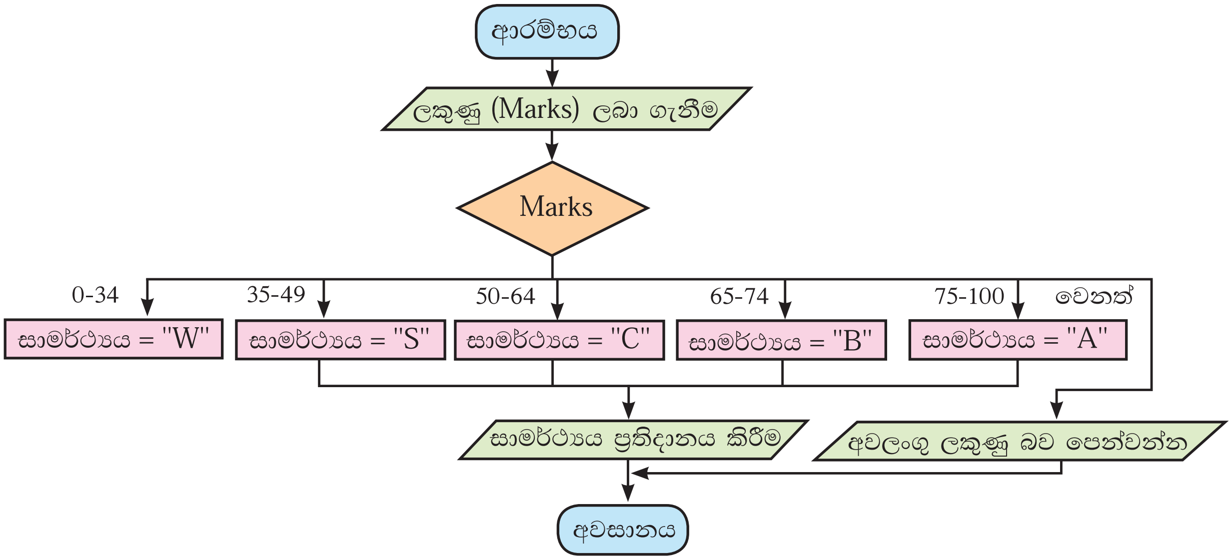
```
Begin
        Input Marks as M
        IF M >= 75 Then
            Grade = ''A''
            ELSE
            IF M >= 65 then
                Grade = ''B''
                ELSE
                IF M >= 50 then
                     Grade = ''C''
                     ELSE
                     IF M >= 35 then
                         Grade = ''S''
                         ELSE
                         Grade = ''F''
                      ENDIF
                ENDIF
             ENDIF
        ENDIF
        Display Grade
End.
```

පැස්කල් ක්‍රමලේඛය

```
program GradeForMarks (input,output);
Var     M: integer;
        Grade: char;
Begin
        Writeln('Enter Marks');
        Read(M);
        If M >= 75 then
           Grade := 'A'
        Else
           If M >= 65 then
        Grade := 'B'
           Else
                       If M >= 50 then
                 Grade := 'C'
             Else
                 If M >= 35 then
                    Grade := 'S'
                 Else
                    Grade := 'F';
        Writeln(''Grade = '', Grade);
End.
```

කේවල විචල්‍යයකට බහු කොන්දේසි ඇති විට **Case Statement** භාවිතය

IF …THEN… ELSE… ENDIF වරණ පාලන ව්‍යුහය නැවත නැවත යොදා ගැනීමට වඩා පහසුවෙන් Case භාවිත කළ හැකි ය.



ගැලීම් සටහන

```
program FindGrade(input,output);
var     Marks : integer;
        Grade: char;
Begin
        Writeln('Enter Marks');
        Read(Marks);
        Case Marks of
                0..34 : Grade := 'W';
                35..49 : Grade := 'S';
                50..64 : Grade := 'C';
                65..74 : Grade := 'B';
                75..100 : Grade := 'A';
        Else
                Writeln('Invalid Marks');
        End;
        if (Marks >= 0) AND (Marks <= 100) then
                Writeln('Grade is ' , Grade);
End.
```

පැස්කල් ක්‍රමලේඛය

1.7 පුනර්කරණ භාවිතයෙන් ක්‍රමලේඛ ලිවීම

පුනර්කරණ වාර ගණන නිශ්චිත ව දන්නා අවස්ථාවල දී පුනර්කරණ ව්‍යුහ භාවිත කරන ආකාරය විමසා බලමු.

i) FOR - DO ව්‍යුහය (ආකාරය 1)

FOR Variable := Value_1 TO Value_2 DO

- මෙහි Variable , Value_1 සහ Value_2 වල දත්ත ප්‍රරූපය නිඛිල සංඛ්‍යා (integer) විය යුතු ය.
- පුනර්කරණය ආරම්භ වීම සඳහා Value_2 හි අගය Value_1 හි අගයට වඩා වැඩි විය යුතු ය.
- පුනර්කරණය Value_1 වලින් ආරම්භ වන අතර Value_2 වලින් අවසන් වේ.
- එබැවින් FOR - DO ව්‍යුහයක් භාවිත කළ හැකි වන්නේ පුනර්කරණය වන වාර ගණන නිශ්චිත ව දන්නේ නම් පමණි.

පුනර්කරණ ව්‍යුහය	ආරම්භක අගය	අවසාන අගය	පුනර්කරණය වන වාර ගණන
FOR X := 1 TO 5 DO	1	5	5
FOR X := 0 TO 4 DO	0	4	5
FOR X := 5 TO 10 DO	5	10	6

උදා : 1 සිට 10 දක්වා අගයන් ප්‍රතිදානය කිරීම

```
Program print10Nos (input,output);
Var     count : integer;
Begin
        For  count := 1 to 10 do
        Writeln(count);
End.
```

මෙහි දී count විචල්‍ය අගය 1 සිට 10 දක්වා වෙනස් වෙමින් ප්‍රතිදානය කරන අතර පුනර්කරණ වාර දහයක් ක්‍රියාත්මක වේ.

ii) FOR - DO ව්‍යුහය (ආකාරය 2)

FOR Variable := Value_1 DOWNTO Value_2 DO

- පුනර්කරණය ආරම්භ වීම සඳහා Value_1 හි අගය Value_2 හි අගයට වඩා වැඩි විය යුතු ය.
- පුනර්කරණය Value_1 වලින් ආරම්භ වන අතර Value_2 වලින් අවසන් වේ.

පුනර්කරණ ව්‍යුහය	ආරම්භක අගය	අවසාන අගය	පුනර්කරණය වන වාර ගණන
FOR X := 10 DOWNTO 5 DO	10	5	6
FOR X := 4 DOWNTO 0 DO	4	0	5

උදා : 10 සිට 1 දක්වා අගයන් ප්‍රතිදානය කිරීම

```
Program printReverse (input,output);

Var     count : integer;
Begin
        For  count := 10  downto 1 do
        Writeln(count);
End.
```

මෙහි දී count විචල්‍ය අගය 10 සිට 1 දක්වා වෙනස් වෙමින් ප්‍රතිදානය කරන අතර පුනර්කරණ වාර දහයක් ක්‍රියාත්මක වේ.

සංඛ්‍යා දහයක එකතුව සහ සාමාන්‍යය සෙවීම

```
program total_avg (input,output);
var     I,num,total : integer;
        avg: real;
Begin
        total := 0;
        for I := 1 to  10 do
        begin
                writeln('Enter Number');
                read(num);
                total := total + num;
        end;
        avg := total/I;
        writeln('Total is ' , total);
        writeln('Average is ',avg);
End.
```

පුනර්කරණය තුළ සංයුක්ත ප්‍රකාශනයකි.

> **වැදගත්**
>
> සංයුක්ත ප්‍රකාශනයක් begin සහ end අතර ලියනු ලැබේ.

පුනර්කරණ වාර ගණන අවිනිශ්චිත අවස්ථාවල දී while do ව්‍යුහය සහ repeat until ව්‍යුහය භාවිත වේ.

i) while do ව්‍යුහය

- පුනර්කරණය ආරම්භයේ දී ම කොන්දේසිය පරීක්ෂා කෙරේ.
- කොන්දේසිය සත්‍ය වන්නේ නම් පමණක් පුනර්කරණය ආරම්භ වේ.
- කොන්දේසිය අසත්‍ය වන්නේ නම් කිසි විටෙක පුනර්කරණය ආරම්භ නො වේ.
- පුනර්කරණය වීම අවසන් වීම සඳහා කොන්දේසිය අසත්‍ය විය යුතු ම ය.
- පුනර්කරණය වන අතර වාරයේ දී කොන්දේසිය අසත්‍ය නොවේ නම් අපරිමිත වාර ගණනක් පුනර්කරණය (infinite loop) වේ.

උදා 1 - while number > 0 do

number විචල්‍යයේ අගය ධන නම් පමණක් පුනර්කරණය සිදු වේ.

උදා 2 -
```
number := 1;
while number <= 10 do
    number := number + 1;
```

- number විචල්‍යයේ ආරම්භක අගය 1 බැවින් කොන්දේසිය සත්‍ය වේ.
- එම නිසා පුනර්කරණය ආරම්භ වේ.
- පුනර්කරණය වන වාරයක් පාසා number හි අගයට 1 ක් එකතු වේ.
- එබැවින් number හි අගය 10 හෝ 10 ට අඩු විට පුනර්කරණය සිදුවේ.
- number විචල්‍යයේ අගය 11 වූ විට පුනර්කරණය වීම නවතී.

ii) Repeat Until ව්‍යුහය

- පුනර්කරණය ආරම්භයේ දී ම කොන්දේසිය පරීක්ෂා නො කෙරේ.
- පුනර්කරණය වීමට ඇති ප්‍රකාශන එක් වරක් ක්‍රියාත්මක වීමෙන් පසු ව කොන්දේසිය පරීක්ෂා කෙරේ.
- කොන්දේසිය අසත්‍ය වන්නේ නම් පමණක් පුනර්කරණය වීම ආරම්භ වේ.
- කොන්දේසිය සත්‍ය වීමත් සමග ම පුනර්කරණය වීම අවසන් වේ.
- පුනර්කරණය වන අතර වාරයේ දී කොන්දේසිය සත්‍ය නොවේ නම් අපරිමිත වාර ගණනක් පුනර්කරණය (Infinite loop) වේ.

උදා 1 - පැස්කල් යන වචනය පුනර්කරණය වීම

```
count := 0;
Repeat
        writeln ('Pascal');
        count := count+1
Until count > 5;
```

- count විචල්‍යයේ ආරම්භක අගය 0 වේ.
- Pascal යන වචනය තිරය මත දර්ශනය වේ.
- count විචල්‍යයේ අගයට 1ක් එකතු වේ.
- count විචල්‍යයේ අගය 5ට විශාල වේ දැයි පරීක්ෂා කෙරේ.
- මෙසේ count විචල්‍යයේ අගය 5 වන තෙක් පුනර්කරණය සිදුවේ.
- countහි අගය 6 වූ විට පුනර්කරණය වීම නවතී.
- පුනර්කරණය නවතින විට Pascal යන වචනය තිරය මත 6 වරක් දර්ශනය වේ.

උදා 2 -

```
sum := 0;
repeat
        sum := sum + 5;
        writeln(sum);
until sum < 50;
```

- sum විචල්‍යයේ ආරම්භක අගය 0 වේ.
- sum හි අගයට 5ක් එකතු වේ.
- sum හි අගය වන 5 තිරය මත දර්ශනය වේ.
- sum විචල්‍යයේ අගය 50 ට අඩු වේ දැයි පරීක්ෂා කෙරේ.
- sum < 50 කොන්දේසිය තෘප්ත වේ. (සත්‍ය වේ.)
- එබැවින් පුනර්කරණය වීම නවතී.

උදා 3 -

```
sum := 0;
repeat
        sum := sum + 5;
        writeln(sum);
until sum >= 50;
```

- sum විචල්‍යයේ ආරම්භක අගය 0 වේ.
- sumහි අගයට 5 ක් එකතු වේ.
- sumහි අගය තිරය මත දර්ශනය වේ.
- sum විචල්‍යයේ අගය 50 ට විශාල හෝ සමාන වේ දැයි පරීක්ෂා කෙරේ.
- sum >= 50 කොන්දේසිය තෘප්ත (සත්‍ය) වන තෙක් පුනර්කරණය වේ.

- පුනර්කරණය නවතින විට sum හි අගය 50 වේ.
- වාර 10 ක් පුනර්කරණය වේ.
- ප්‍රතිදානය ලෙස 5 සිට 50 දක්වා ඇති 5 හි ගුණාකාර ලබා දෙයි.

1.8 නීඩිත පාලන ව්‍යුහවල (Nested Control Structures) අවශ්‍යතාව

වරණය හෝ පුනර්කරණය එකිනෙක වෙනස් පාලන ව්‍යුහ දෙකක් වුව ද ගැටලුවේ ස්වභාවය මත වරණයකින් පසු ව පුනර්කරණයක් සිදුවීම හෝ සිදු නොවීම විය හැකි ය. එමෙන් ම පුනර්කරණය සිදුවන අතරතුර වරණයකට අදාළ ව එකිනෙකට වෙනස් ක්‍රියාවලි සිදුවිය හැකි ය. මෙවැනි තත්ත්ව යටතේ ක්‍රමලේඛකරණය සඳහා නීඩිත පාලන ව්‍යුහ යොදා ගැනීමට අවශ්‍ය වේ.

1.8.1 වරණය තුළ පුනර්කරණ යොදා ගැනීම

වරණයක කොන්දේසියක් තෘප්ත වීම හෝ නොවීම හෝ අනුව පුනර්කරණයක් සිදුවිය හැකි ය.

උදා - පරිශීලකගේ තේරීම අනුව ප්‍රතිදානය, ආරෝහණ හෝ අවරෝහණ සංඛ්‍යා අනුක්‍රමයක් ලැබීම

```
program orderNos(input,output);
var     num:integer;
        cho:char;
begin

   writeln('Select Assending(A) or Desending(D)');
   read(cho);
   if cho =  'A' then
     begin
         writeln('Asending Order');
         for num := 1 to 6 do
            writeln(num);
     end;
   if cho = 'D' then
      begin
         writeln('Desending Order');
         for num := 6 downto 1 do
            writeln(num);
      end;
end.
```

1.8.2 පුනර්කරණය තුළ වරණය යොදා ගැනීම

පුනර්කරණය සිදු වන අතරතුර දී වරණයක් සිදු කරන ආකාරය සලකා බලමු.

උදා - පරිශීලක විසින් ආදානය කරනු ලබන සංඛ්‍යා ඔත්තේ සංඛ්‍යා ද, ඉරට්ටේ සංඛ්‍යා ද, යන්න හඳුනා ගැනීම හා ඔත්තේ හා ඉරට්ටේ සංඛ්‍යා ප්‍රමාණ වෙන වෙන ම ගණනය කිරීම

```
program rep_sel(input,output);
var num,rem,count,e_count,o_count:integer;
begin
 e_count := 0, o_count := 0;
    for count := 1 to 10 do
        begin
            writeln('Enter Number');
            read(num);
            rem := num mod 2;
            if rem = 0 then
              begin
                  writeln('Even number');
                   e_count := e_count + 1;
              end
            else
               begin
                   writeln('Odd number');
                   o_count := o_count + 1;
               end;
        end;
    writeln(e_count,'Even Number/s');
    writeln((o_count ,'Odd Number/s');
end.
```

1.9 අරාව (Array) භාවිතය

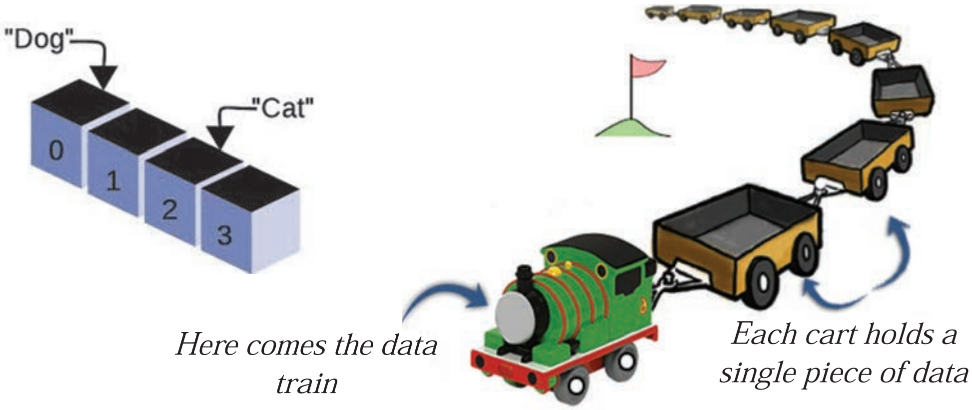
ක්‍රමලේඛනයේ දී මතකය තුළ දත්ත තැන්පත් කිරීමට විචල්‍යය භාවිත කිරීම අනිවාර්ය වේ. එමෙන් ම එම විචල්‍යයක් සඳහා සුදුසු දත්ත ප්‍රරූපයක් පැවතිය යුතු ය. එක ම දත්ත ප්‍රරූපයට අයත් දත්ත තැන්පත් කිරීමට එකිනෙකට වෙනස් නම් වලින් යුත් වෙනස් විචල්‍ය ද සමූහයක් අවශ්‍ය වේ.

උදා - පූර්ණ සංඛ්‍යා 5 ක් මතකය තුළ තැන්පත් කිරීම සඳහා විචල්‍ය 5 ක් අවශ්‍ය වේ. එවැනි විචල්‍ය භාවිතයට පෙර පහත දැක්වෙන ආකාරයෙන් හඳුන්වා දිය යුතු ය.

Var p, q, r, s, t : integer; හෝ
n1, n2, n3, n4, n5 : real;

1.9.1 අරාවන් භාවිතයේ අවශ්‍යතාව

එක ම ප්‍රරූපයට අයත් දත්ත එක ම නමක් යොදා ගනිමින් මතකය තුළ තැම්පත් කිරීමට අරාව භාවිත කෙරේ. එබැවින් එක් එක් දත්තය සඳහා වෙන වෙන ම විචල්‍ය නාම වෙනුවට තනි විචල්‍ය නාමයක් යටතේ අවශ්‍ය ප්‍රමාණයකට දත්ත තැන්පත් කිරීමට අරාවක් භාවිතයෙන් හැකි වේ.



1.9.2 ඒකමාන අරාවක් අර්ථ දැක්වීම

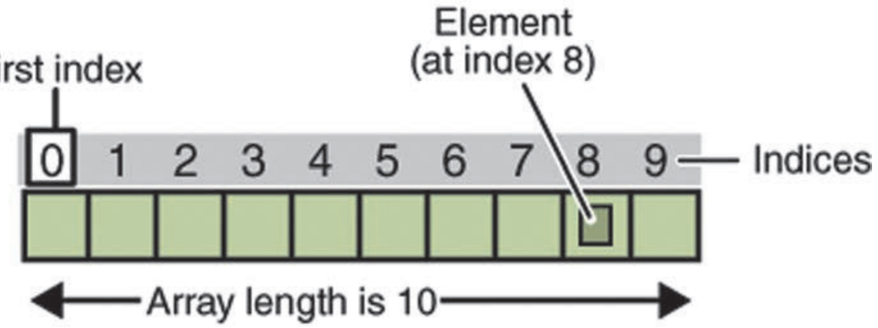
අනුක්‍රමික ව නිශ්චිත ප්‍රමාණයක එක ම ප්‍රරූපයකට අයත් දත්ත තැන්පත් කිරීම සඳහා භාවිත කරන දත්ත ව්‍යුහයකි. අරාවක් මගින් යාබද මතක අවකාශ සමූහයක් වෙන් කර ගනී.

ඒක-මාන අරාවක් පහත දැක්වෙන අයුරින් හඳුන්වා දෙනු ලබයි.

Var Name_of_Array : array [first value .. last value] of data type

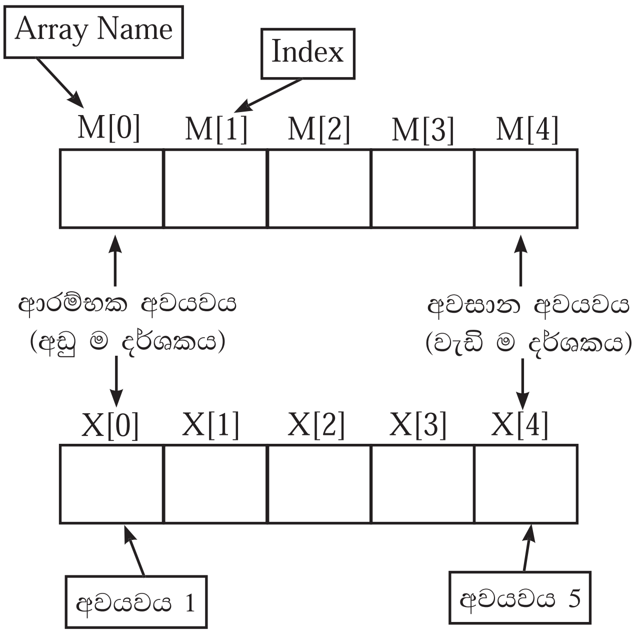
උදා - var marks : array [0..9] of integer;

- මෙමගින් පූර්ණ සංඛ්‍යාත්මක අගයන් 10 ක් ඇතුළත් කළ හැකි marks නම් වූ අරාව සෑදේ.



1.9.3 අරාවක ලක්ෂණ

- අරාවක කොටස් එනම් අවයව (Element) යාබද ව පිහිටයි.
- අරාවක දර්ශකය (Index) එනම්, අනුපිළිවෙළ අංකය අරාවේ නම සමග කොටු වරහන් තුළ දක්වනු ලැබේ.



උදා -

Var M : Array[0..4] of integer; වන අරාවේ ප්‍රමාණය 5කි.

එය M[0] සිට M[4] දක්වා අවයව 5 කින් සමන්විත වේ.

කොටු වරහන තුළ දර්ශකය දක්වනු ලබයි.

අරාව හඳුන්වා දෙන ක්‍රමය අනුව එහි දර්ශක පිහිටන ස්ථානය වෙනස් වේ.

උදා - Var X : Array[1..5] of integer;

එක ම ප්‍රරූපයකට අයත් දත්ත පමණක් අරාවේ තැන්පත් කළ හැකි ය.

අරාවේ ඕනෑ ම අවයවයක් වෙත අහඹු ලෙස පිවිසීමට හැකි ය. එබැවින් පුනර්කරණ ව්‍යුහයක් මගින් අරාවක් පහසුවෙන් පාලනය කළ හැකි ය.

උදා - සිසුන් 40 දෙනකුගේ ගණිත ලකුණු අරාවකට ඇතුළත් කිරීම

```
var     maths : array[0..39] of integer;
        i,marks : integer;
for i := 0 to 39 do
        begin
                writeln('Enter marks');
                read(marks);
                maths[i] := marks;
        end;
```

1.9.4 අරාවකට අගයන් පැවරීම (Assign values to an array)

අරාවක අවයවවලට වෙන වෙන ම අගයයන් පවරනු ලැබේ.

පූර්ණ සංඛ්‍යා ඇතුළත් කළ හැකි අවයව 5කින් සමන්විත num අරාව සලකමු.

	num[0]	num[1]	num[2]	num[3]	num[4]
var num : array[0..4] of integer;					
num[0] := 45;					
num[2] := 36,num[4] := 60;	45		36		60
num[1] := num[4] + 15; num[3] := num[0] + num[2]	45	75	36	81	60

1.9.5 අරාවක අගයන් ප්‍රකාශයට පත් කිරීම (Display values in an array)

අරාවක අගයන් ප්‍රකාශයට පත් කිරීම එහි අවයව මගින් සිදු කෙරේ.

writeln (num[3]); - 4 වන අවයවය ප්‍රකාශයට පත් කිරීම (81)

writeln (num[1], num[4]); - 2, 5 වන අවයව ප්‍රකාශයට පත් කිරීම (36, 60)

for x := 0 to 3 do - අරාවේ පළමු අවයව 4 ප්‍රකාශයට පත් කිරීම (45, 75, 36, 81)

writeln (num[x]);

for x := 2 to 4 do - අරාවේ 3, 4, 5 යන අවයව 3 ප්‍රකාශයට පත් කිරීම (36, 81, 60)

writeln (num[x]);

for x := 0 to 4 do - අරාවේ සියලු ම අවයව ප්‍රකාශයට පත් කිරීම (45, 75, 36, 81, 60)

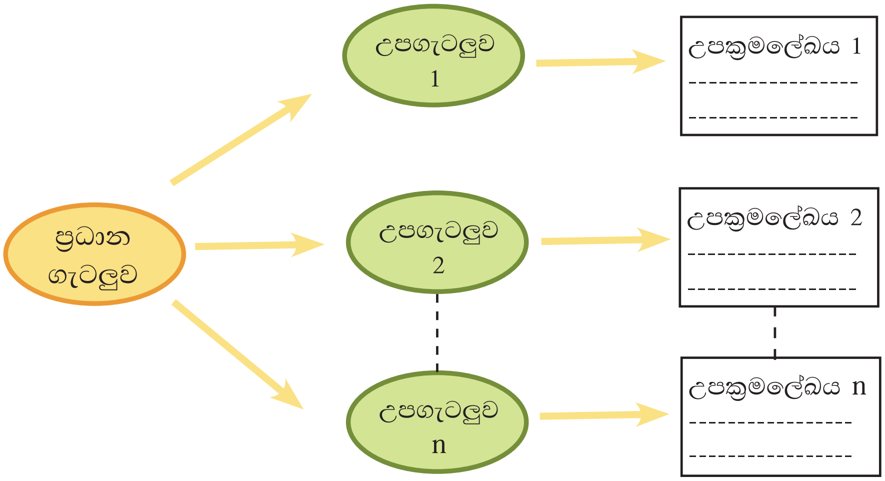
writeln (num[x]);

උදා - පන්තියක සිසුන් 35 දෙනෙකුගේ තොරතුරු හා සන්නිවේදන තාක්ෂණ විෂයයට ලබා ගත් ලකුණු ඇතුළත් කර වැඩි ම ලකුණ සහ ලකුණුවල සාමාන්‍ය අගය ගණනය කිරීම.

```
program ictMarks(input,output);
var marks:array[0..34] of integer;
   i,tot,max:integer;
   avg:real;
begin
    for i := 0 to 34 do
       begin
          writeln('Enter Marks');
          read(marks[i]);   (* Read Marks to array *)
          tot := tot + marks[i]; (* Add marks *)
       end;
    avg := tot/35;
    max := marks[0];
    for i := 1 to 34 do
       if marks[i] > max then max := marks[i];
    writeln('Maximum marks = ', max);
    writeln('Average marks = ',avg);
end.
```

1.10 උපක්‍රමලේඛ භාවිතය

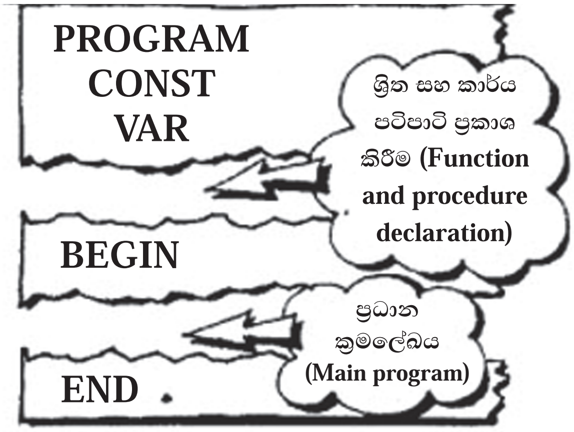
ක්‍රමලේඛයක ක්‍රියාවලි ප්‍රමාණය වැඩිවීමත් සමග ම එය සංකීර්ණ වන නිසා කියවා තේරුම් ගැනීම, ක්‍රමලේඛය නඩත්තු කිරීම වැනි කටයුතු අපහසු වේ. එබැවින් ක්‍රමලේඛ ලිවීමේ දී හැකිතාක් දුරට උපක්‍රමලේඛ ලෙස ලිවීම සාර්ථක ක්‍රමයකි.



1.10.1 උපක්‍රමලේඛ වර්ග

ප්‍රධාන ක්‍රමලේඛය හා සම්බන්ධ උපක්‍රමලේඛ මගින් ප්‍රධාන ක්‍රමලේඛය වෙත ප්‍රතිදානයක් ආපසු ලබා දීමට හැකි සහ නොහැකි වශයෙන් උපක්‍රමලේඛ වර්ග දෙකක් ඇත. ප්‍රතිදානයක් ආපසු ලබා දිය හැකි උපක්‍රමලේඛ ශ්‍රිත (Function) ලෙස ද, ප්‍රතිදානයක් ආපසු ලබා දිය නොහැකි උපක්‍රමලේඛ කාර්ය පටිපාටිය (Procedure) ලෙස ද හැඳින්වේ.

1.10.2 උපක්‍රමලේඛ හඳුන්වා දීම



ප්‍රධාන ක්‍රමලේඛය ආරම්භ කිරීමට ප්‍රථමයෙන් ශ්‍රිත සහ කාර්ය පටිපාටි හඳුන්වා දිය යුතු ය. ප්‍රධාන ක්‍රමලේඛය තුළ දී උපක්‍රමලේඛවලට ඇමතිය (Calling the function or Procedure) යුතු ය.

කාර්ය පටිපාටියක් හඳුන්වා දීම සඳහා නිවැරදි කාරක රීතිය (Syntax) පහත දැක්වේ.

Procedure Name_of_Procedure (name_of_variable: data type);

උදා - වෘත්තයක වර්ගඵලය සෙවීම සඳහා කාර්ය පටිපාටිය

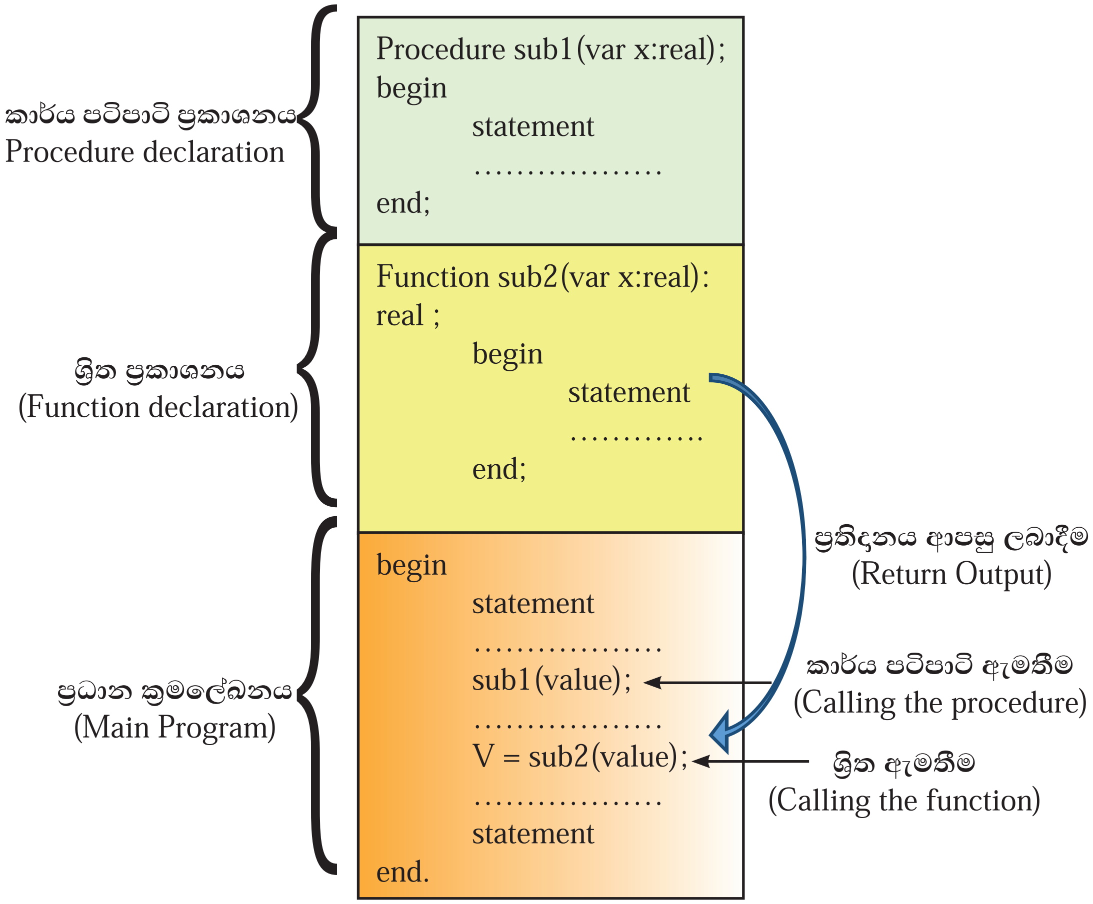
Procedure calculateArea (var radius : real);

ශ්‍රිතයක් හඳුන්වා දීම සඳහා නිවැරදි කාරක රීතිය (Syntax) පහත දැක්වේ.

Function Name_of_Function(name_of_variable : data type) : data type ;

උදා - වෘත්තයක වර්ගඵලය සෙවීම සඳහා ශ්‍රිතය

Function calculateArea (var radius : real): real;



උදා - වෘත්තයක වර්ගඵලය සහ පරිධිය ගණනය කිරීමේ ක්‍රමලේඛය සලකමු.

1. කාර්ය පටිපාටි යොදා ගෙන ගොඩනගන ලද ක්‍රමලේඛය

```
program procedure_circle(input,output);
const pie = 22/7;
var radius:real;
procedure getData(var radius: real);
        begin
            writeln('Enter Radius');
            read(radius);
        end;
procedure processArea(var radius:real);
        var area:real;
        begin
            area := pie * radius * radius;
            writeln('Area = ',area);
        end;
procedure processCircumferance(var radius:real);
        var circum:real;
        begin
            circum := 2 * pie * radius;
            writeln('Circumferance = ',circum);
        end;
begin
    getData(radius);
    processCircumferance(radius);
    processArea(radius);
end.
```

2. ශ්‍රිත යොදා ගෙන ගොඩ නගන ලද ක්‍රමලේඛය

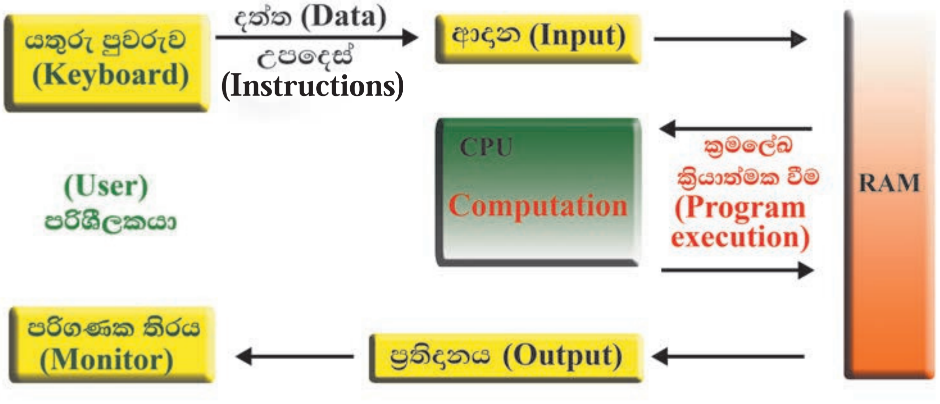
```
program function_circle(input,output);
const pi = 22/7;
var radius:real;
function processArea(var radius:real):real;
        var area:real;
        begin
```

```
            area := pi * radius * radius;
            processArea := area;
        end;
function processCircumferance(var radius:real):real;
        var circum:real;
        begin
           circum := 2 * pi * radius;
           processCircumferance := circum;
        end;
begin
    writeln('Enter Radius');
            read(radius);
    writeln('Circumferance = ',processCircumferance(radius));
    writeln('Area = ', processArea(radius));
end.
```

1.11 ක්‍රමලේඛ භාෂාවල පරිණාමය

1.11.1 ක්‍රමලේඛ භාෂාවක අවශ්‍යතාව

ක්‍රමලේඛයක් යනු පරිගණකය විසින් කළ යුතු විශේෂ කාර්යයක් ඉටුකරන ආකාරය දැක්වෙන උපදෙස් අනුක්‍රමයකි. මෙම උපදෙස් ලබා දීම සඳහා භාෂාවක් අවශ්‍ය වේ.



පරිගණකයේ ක්‍රමලේඛයක් ක්‍රියාත්මක වන ආකාරය

1.11.2 පහළ තලයේ භාෂා (Low level languages)

යන්ත්‍ර භාෂාව (Machine language)

පරිගණකය තුළ සෘජු ව ම ක්‍රියාත්මක කළ හැකි භාෂාවකි. උපදෙස් ලබා දීම සඳහා 0 සහ 1 යන ද්විමය සංඛ්‍යාංක (බිටු) යොදා ගන්නා ලදි. එබැවින් යන්ත්‍ර භාෂාවෙන් ලියන ලද ක්‍රමලේඛයක් සෘජු ව ම සකසනයට ධාවනය කළ හැකි විය.

යන්ත්‍ර භාෂාවෙන් ලියන ලද ක්‍රමලේඛයක ලක්ෂණ

- ක්‍රියාත්මක වීම ඉතා ම වේගවත් වීම
- භාෂා පරිවර්තක වැඩසටහන් අවශ්‍ය නොවීම
- යන්ත්‍රය මත යැපීම (එක් පරිගණකයකට ලියන ලද ක්‍රමලේඛයක් වෙනත් පරිගණකයක ධාවනය නොවීම)
- 0 සහ 1 පමණක් භාවිතයෙන් ලියා ඇති නිසා මිනිසාට තේරුම් ගැනීම සංකීර්ණ වීම

එසෙම්බ්ලි භාෂාව (Assembly language)

යන්ත්‍ර භාෂාවෙන් 0 සහ 1 මත පදනම් ව ලියන ලද විධාන වෙනුවට සරල සංකේත නාම භාවිත කර එසෙම්බ්ලි භාෂාව නිර්මාණය කර ඇත.

එසෙම්බ්ලි භාෂාවෙන් ලියන ලද ක්‍රමලේඛයක ලක්ෂණ

- ක්‍රියාත්මක වීම යන්ත්‍ර භාෂා ක්‍රමලේඛයකකට සාපේක්ෂ ව වේගවත් බව අඩු ය.
- ඇසෙම්බ්ලර් නම් වූ භාෂා පරිවර්තක වැඩසටහන මගින් යන්ත්‍ර භාෂා උපදෙස් බවට පරිවර්තනය කළ යුතු වීම
- යන්ත්‍රය මත යැපීම (එක් පරිගණකයකට ලියන ලද ක්‍රමලේඛයක් වෙනත් පරිගණකයක ධාවනය නොවීම)
- සංකේත යොදා ගැනීම නිසා තේරුම් ගැනීම යන්ත්‍ර භාෂාවට සාපේක්ෂ ව සරල වීම

එසෙම්බ්ලි භාෂාවෙන් ලියන ලද ක්‍රමලේඛයක් (A program written in Assembly language) → ඇසෙම්බ්ලර් (Assembler) → යන්ත්‍ර භාෂා විධාන (Machine language instructions)

1.11.3 ඉහළ තලයේ භාෂා (High level languages)

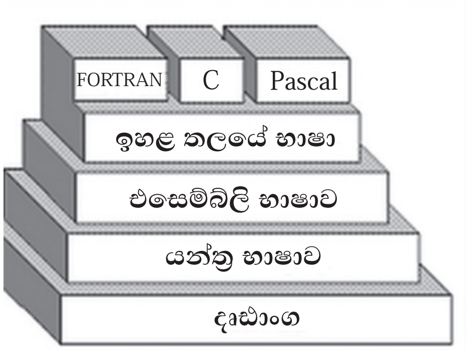
ඉංග්‍රීසි භාෂාවේ සරල වචන යොදා ගනිමින් ක්‍රමලේඛකයාට වඩා පහසුවෙන් තේරුම් ගැනීමට හැකි වන පරිදි නිර්මාණය කරන ලද භාෂා ඉහළ තලයේ භාෂා ලෙස හැඳින්වේ.

ඉහළ තලයේ භාෂා සඳහා උදාහරණ

FORTRAN, BASIC, COBOL, PASCAL, C

ඉහළ තලයේ භාෂාවෙන් ලියන ලද ක්‍රමලේඛයක ලක්ෂණ

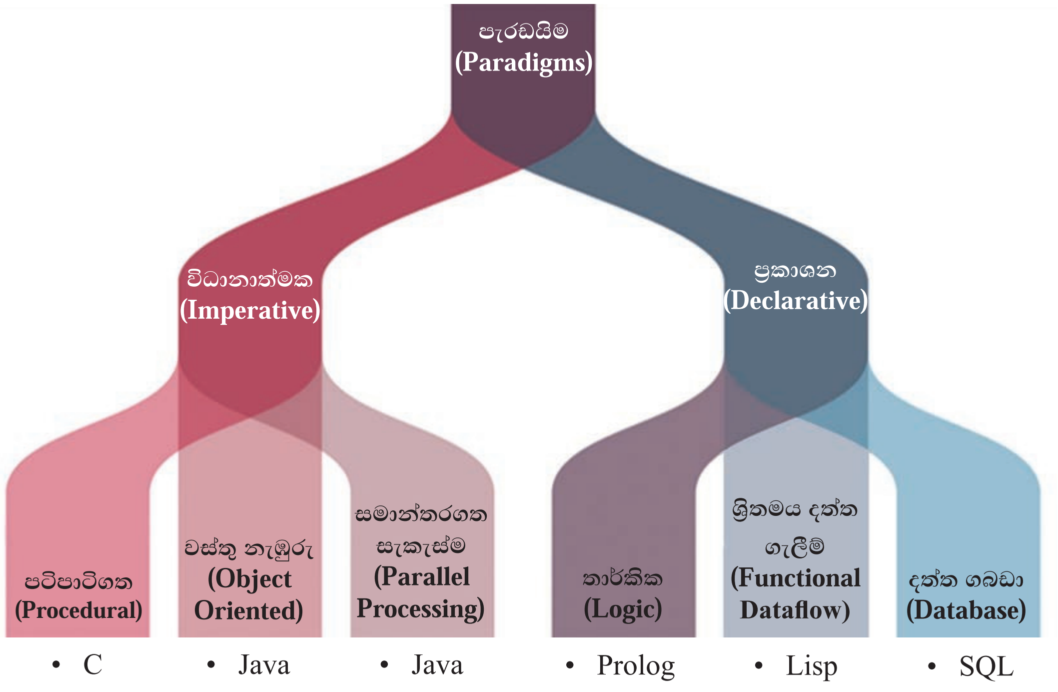
- තේරුම් ගැනීම පහසු ය.
- ධාවනය කිරීමට ප්‍රථමයෙන් යන්ත්‍ර භාෂා උපදෙස් බවට පරිවර්තනය කළ යුතු ය.
- යන්ත්‍රය මත යැපෙන භාෂාවක් නො වේ.



1.11.4 ක්‍රමලේඛ භාෂා වර්ග

ක්‍රමලේඛය කිරීම යනු කිසි යම් කාර්යයක් කරන ආකාරය පිළිබඳ ව පරිගණකයට උපදෙස් දෙනු පිණිස පරිගණක ක්‍රමලේඛකයෙක් විසින් සකසනු ලබන නිර්මාණශීලී ක්‍රියාවලියකි. කිසි යම් ගැටලුවකට විසඳුමක් දෙනු පිණිස කුමන කාර්යයක් කළ යුතු ද යන්න ගැන පරිගණකයට උපදෙස් දීමට සකස් කළ උපදෙස් මාලාව ක්‍රමලේඛයක් ලෙස හැඳින්විය හැකි ය.

ක්‍රමලේඛන ක්‍රියාවලියට ප්‍රවේශ වීම පිණිස විකල්ප ප්‍රවේශ ගණනාවක් ඇත. ඒවා ක්‍රමලේඛන පැරඩයිම (Paradigms) ලෙස හැඳින්වේ. ක්‍රමලේඛන භාවිත කොට විශේෂ ගැටලුවලට විසඳුම් ගොඩනැගීම සඳහා මූලික වශයෙන් වෙනස් ආකාර ප්‍රවේශ, විවිධ පැරඩයිම මගින් නියෝජනය වේ. බහුතර ක්‍රමලේඛන භාෂා එක පැරඩයිම වර්ගයක් යටතට ගැනෙන නමුත් සමහර භාෂාවල විවිධ පැරඩයිමවලට අයත් මූලිකාංග දැකිය හැකි ය.



ක්‍රමලේඛ පැරඩයිම

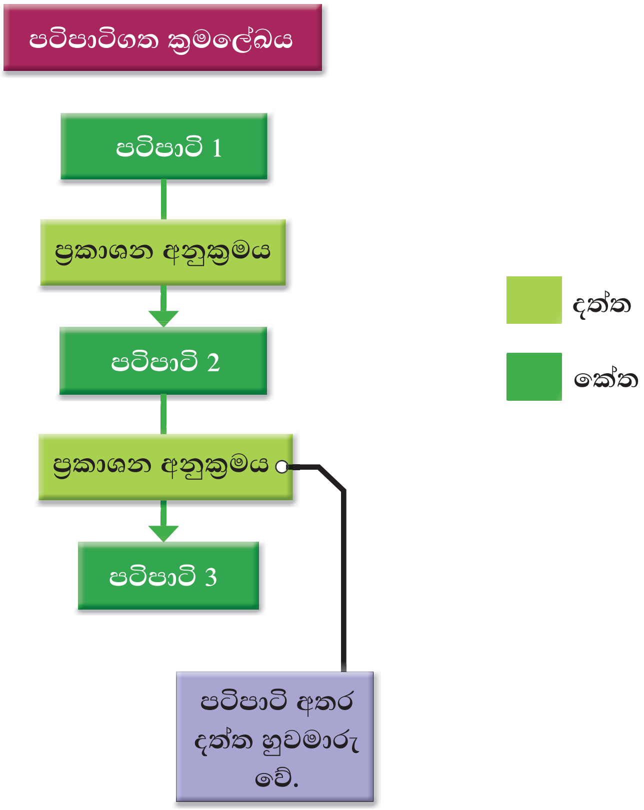
විධානාත්මක	ප්‍රකාශන		වස්තු නැඹුරු
	ශ්‍රිතමය ක්‍රමලේඛ	තාර්කික ක්‍රමලේඛ	
Algol Cobol PL/1 Ada C Modula - 3	Lisp Haskell ML Miranda APL	Prolog	Smalltalk Simula C++ Java

අද වන විට ක්‍රමලේඛන භාෂා විශාල ප්‍රමාණයක් බිහි වී ඇත. ඒවා කාලයත් සමග ඇගයීමකට ලක් කළ විට පහත දැක්වෙන අයුරින් වෙනස්කම් දැකිය හැකි ය.

පටිපාටිගත (Procedural) හා ප්‍රකාශන (Declarative) ක්‍රමලේඛ අතර වෙනස

පටිපාටිගත භාෂාවක් යනු ඉතා හොඳින් ව්‍යුහගත වූ පියවර මාලාවකින් සහ පටිපාටිවලින් (Procedural) සමන්විත පරිගණක ක්‍රමලේඛන භාෂා වර්ගයකි. එහි ක්‍රමවත් පියවරවලින් සමන්විත ප්‍රකාශන (Statements) අඩංගු වේ.

උදා -



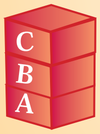
ඔබ ඉගෙනගත් Pascal ක්‍රමලේඛයේ පටිපාටිගත ක්‍රමලේඛයේ ලක්ෂණ ඇත.

ප්‍රකාශන ක්‍රමලේඛයක් යනු පාලන ගැලීම විස්තර කිරීමකින් තොරව, ගණනය කිරීමේ තාර්කික න්‍යාය විදහා දක්වන, පරිගණක ක්‍රමලේඛවල ව්‍යුහය සහ අවයව ගොඩනංවන ආකාරයකි. මෙම නිර්මාණශීලී ආකාරය භාවිත කෙරෙන බොහෝ නිර්මාණ මගින් අතුරු ප්‍රතිඵල අවම කිරීමට හෝ තුරන් කිරීමට හෝ උත්සාහ දරනු ලැබේ. එය කරනු ලබන්නේ ප්‍රාථමික ක්‍රමලේඛ භාෂාවල මෙන් ගැටලුව විසඳීමට කටයුතු කරන්නේ කෙසේ දැයි විස්තර කරනවාට වඩා ගැටලු විසඳීමට කුමන ක්‍රමලේඛය සකස් කළ යුතු දැයි විස්තර කිරීමෙනි. එසේ වුව ද එමගින් කෙලෙස සිදුවන්නේ ද යන්න විස්තර නොවේ. එනම් පරිගණකය දැනුම් දෙන්නේ ගැටලුව කුමක්ද යන්න පමණක් වන අතර එය විසඳන ආකාරය නොවේ. ආදානය කරන ගැටලුවට අවශ්‍ය ව විසඳුම් පරිගණකය මගින් සොයාදෙයි. මෙය විස්තරාත්මක පියවර වශයෙන් ඇල්ගොරිතම ක්‍රියාවට නැංවෙන, පටිපාටිගත ක්‍රමලේඛවලට ප්‍රතිවිරුද්ධ වේ. ප්‍රකාශන ක්‍රමලේඛ කෘතිම බුද්ධිය මත පදනම් වී ඇත.

වැදගත්

පටිපාටිගත ක්‍රමලේඛ
(Procedural Paradigm)

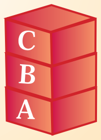
විසඳුම සොයන ආකාරය පැවසීම
Saying how you achieve it



1. A කැටය තබන්න.
2. A කැටය මත B කැටය තබන්න.
3. C කැටය B කැටය මත තබන්න.

ප්‍රකාශන ක්‍රමලේඛ
(Declarative Paradigm)

අවශ්‍ය දේ පැවසීම
Saying what you want



කැට 3කින් සමන්විත කණුවක්

ව්‍යුහගත (Structured) සහ වස්තු නැඹුරු (Object oriented) ක්‍රමලේඛ සංසන්දනය

ව්‍යුහගත ක්‍රමලේඛ යනු තාර්කික ක්‍රමලේඛන පැරඩයිමයක් වන අතර එය වස්තු නැඹුරු ක්‍රමලේඛවල පූර්ව අවස්ථාවකි. ව්‍යුහගත ක්‍රමලේඛ පැරඩයිම මගින් ක්‍රමලේඛය තේරුම් ගැනීම සහ නවීකරණය සඳහා පහසුකම් සලසයි. පද්ධතිය සංරචකමය උප පද්ධතිවලට බෙදා ඇති අතර ඉහළ සිට පහළට (Top - down) ප්‍රවේශ ඇත.

ක්‍රමලේඛයේ කේතවලට තාර්කික විධාන පැවරීමේ තාක්ෂණයක් මෙහි ඇත. එබැවින් එහි ක්‍රියාවලිය පහසු කරයි. විශාල ක්‍රමලේඛ කුඩා ඒකකවලට බෙදා වෙන් කර ඇති අතර එම කොටස් ක්‍රමානුකූල ව එකිනෙක අතර බැඳී පවතී.

ව්‍යුහගත ක්‍රමලේඛ

පරිශීලකට දත්ත ඇතුළත් කරන ලෙස දැනුම් දෙයි.

↓

පරිශීලක ඇතුළත් කළ දත්ත ලබා ගනී.

↓

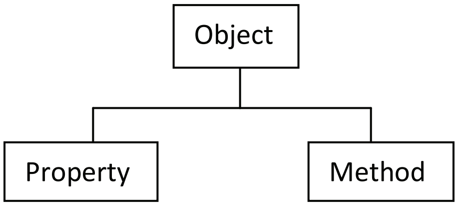
දත්ත ක්‍රියාවලියට භාජනය කරයි.

↓

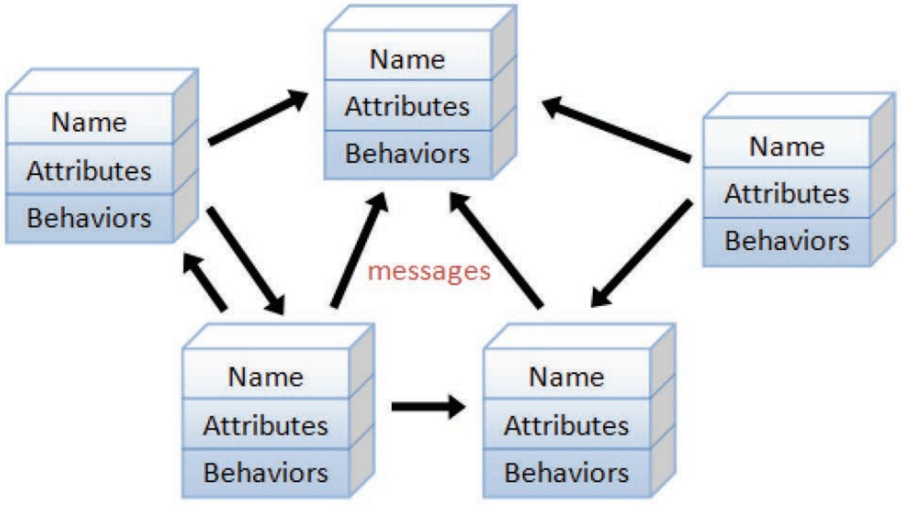
පරිශීලකට ප්‍රතිඵලය ලබා දෙයි.

ඔබ ඉගෙනගත් Pascal ක්‍රමලේඛයේ ව්‍යුහගත ක්‍රමලේඛ ලක්ෂණ ද පවතී.

වස්තු නැඹුරු පරිගණක ක්‍රමලේඛ යනු වස්තු (Objects) යන සංකල්පය මත පදනම් වූ ක්‍රමලේඛ පැරඩයිම වේ. වස්තු, දත්ත (data) සහ ක්‍රමවලින් (Methods) සමන්විත වේ. ක්‍රම (Methods) යනු දත්ත හැසිරවීම සඳහා පවතින ක්‍රියාමාර්ග (Procedure) ස්වරූපයෙන් පවතින කේත වේ.



පන්ති (Class) යනු වස්තු නැඹුරු ක්‍රමලේඛවල මූලික තැනුම් ඒකකයයි. පන්ති මගින් දත්ත සහ පන්තියේ අවස්ථා හා සම්බන්ධ හැසිරීම් විස්තර කරයි. පන්ති භාවිතයෙන් එකම ආකාරයකට අයත්වන වස්තු නිර්මාණය කළ හැකි ය.



වස්තු නැඹුරු ක්‍රමලේඛ එකිනෙක අන්තර් ක්‍රියාකරන්නා වූ වස්තුන්ගෙන් සමන්විත ය.

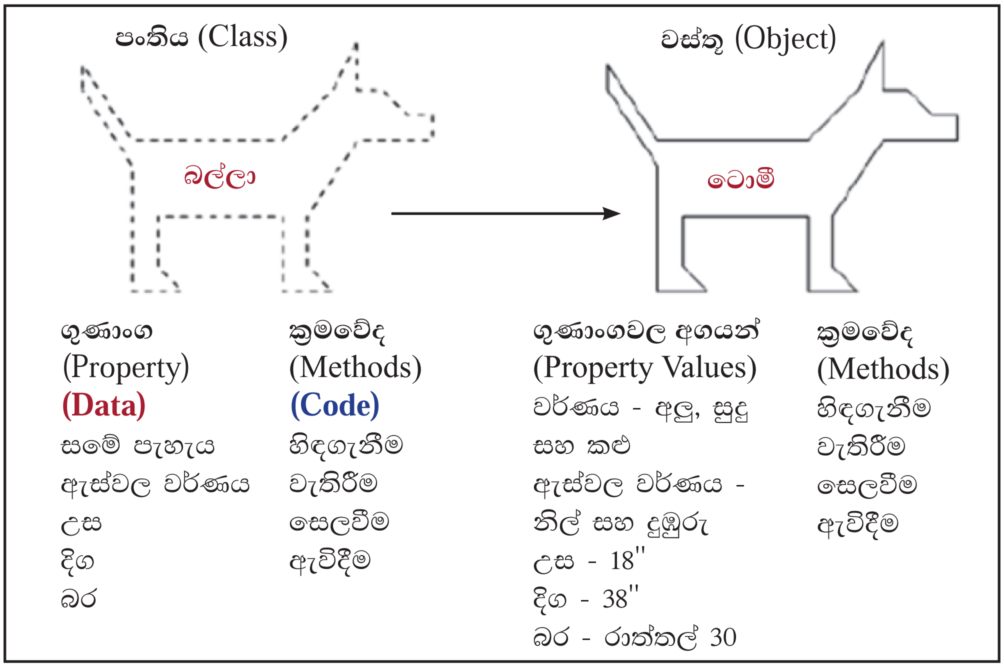
පුස්තකාල තොරතුරු පද්ධතිය
(Library Information Systems)

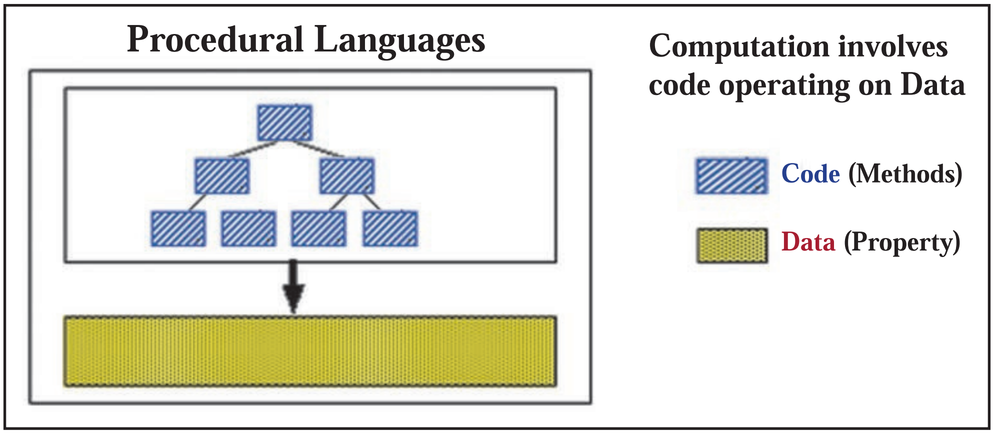
ව්‍යුහගත පිවිසීම (Structured Approach)
ශ්‍රිත සහ ක්‍රියාපටිපාටි වලින් සමන්විත ය.
(Decompose by functions or processes)

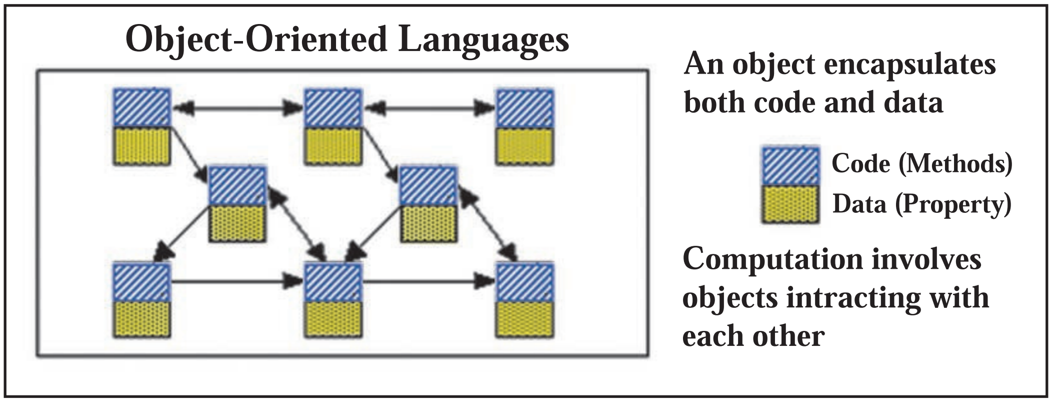
පද්ධතිය **(System)**
- වාර්තා සැකසීම (Record Loans)
- සම්පත් එකතු කිරීම (Add Resources)
- ගොනු වාර්තා කිරීම (Report Fines)

වස්තු නැඹුරු පිවිසීම (Object Approach)
වස්තු සහ ක්‍රමවේදවලින් සමන්විත ය.
(Decompose by objects or concepts)

- සූචිය (Catalog)
- පුස්තකාලාධිපති (Librarian)
- පොත් (Book)
- පුස්තකාලය (Library)


පංතිය (Class)
වස්තු (Object)
බල්ලා
ටොමී
ගුණාංග
(Property)
(Data)
සමේ පැහැය
ඇස්වල වර්ණය
උස
දිග
බර
ක්‍රමවේද
(Methods)
(Code)
හිඳගැනීම
වැතිරීම
සෙලවීම
ඇවිදීම
ගුණාංගවල අගයන්
(Property Values)
වර්ණය - අලු, සුදු සහ කළු
ඇස්වල වර්ණය - නිල් සහ දුඹුරු
උස - 18''
දිග - 38''
බර - රාත්තල් 30
ක්‍රමවේද
(Methods)
හිඳගැනීම
වැතිරීම
සෙලවීම
ඇවිදීම


Procedural Languages
Computation involves code operating on Data
Code (Methods)
Data (Property)


Object-Oriented Languages
An object encapsulates both code and data
Code (Methods)
Data (Property)
Computation involves objects intracting with each other

ක්‍රියාකාරකම

Car නම් පන්තිය (Class) සැලකූ විට එහි වස්තු (Objects), ගුණාංග (Properties) සහ ක්‍රමවේද (Methods) මොනවාද යන්න හඳුනාගන්න.

ක්‍රමලේඛනය (Programming) සහ පිටපත් ලේඛනය (Scripting)

ක්‍රමලේඛන භාෂාවල සාමාන්‍යයෙන් පද යෝජනය (Syntax) සම්බන්ධයෙන් දැඩි නීති රීති පවතී. නිතර ම පාහේ සම්පාදනය කිරීම අවශ්‍ය වේ. ක්‍රමලේඛ භාෂා සම්පාදනය (Compile) කළ යුතු වේ. පිටපත් ලේඛන භාෂා සාමාන්‍යයෙන් වෙනත් යෙදවුමක් (උදා - වෙබ් බ්‍රවුසරයක් හෝ වෙබ් සර්වරයක්) සමග බොහෝ විට ක්‍රියාත්මක වන කේතයක් නිෂ්පාදනය කෙරේ. එමෙන් ම පිටපත් ලේඛන භාෂා අර්ථ වින්‍යාශ (Interprit) කළ යුතු ය. නවකයන්ගෙන් අතිශය බහුතරයක් පිටපත් රචන භාෂාව භාවිත කිරීම වඩා පහසු යැයි සිතති.

JavaScript සහ PHP යන දෙක ම පිටපත් ලේඛන භාෂා වේ.

ක්‍රියාකාරකම

පහත සඳහන් ක්‍රමලේඛ පැරඩයිමවල වෙනස්කම් සංසන්දනය කරන්න.

- පටිපාටිගත හා ප්‍රකාශන (Procedural vs Declarative)
- ව්‍යුහගත හා වස්තු නැඹුරු (Structured vs Object oriented)
- ක්‍රමලේඛ හා පිටපත් ලේඛනය (Programming vs Scripting)

1.11.5 භාෂා පරිවර්තක වැඩසටහන්

යන්ත්‍ර භාෂාවෙන් හැර වෙනත් ඕනෑ ම භාෂාවකින් ලියන ලද ක්‍රමලේඛ (වස්තු කේත) ධාවනය කිරීමට ප්‍රථමයෙන් යන්ත්‍ර භාෂා උපදෙස් (යන්ත්‍ර කේත) බවට පරිවර්තනය කළ යුතු ය.

ඇසෙම්බ්ලි භාෂාවෙන් ලියන ලද ක්‍රමලේඛයක් ඇසෙම්බ්ලර් නම් වූ භාෂා පරිවර්තක වැඩසටහන මගින් යන්ත්‍ර භාෂා උපදෙස් බවට පරිවර්තනය කරනු ලැබී ය.

ඉහළ තලයේ භාෂාවක් යොදා ගනිමින් ලියන ලද ක්‍රමලේඛයක් යන්ත්‍ර භාෂා උපදෙස් බවට පරිවර්තනය කිරීම සඳහා වැඩසටහන් (මෘදුකාංග) දෙකක් යොදා ගැනේ.

1. අර්ථ වින්‍යාසක (Interpreter)
2. සම්පාදක (Compiler)

අර්ථ වින්‍යාසක (Interpreter)

ඉහළ තලයේ පරිගණක භාෂාවකින් ලියන ලද ක්‍රමලේඛයක් ක්‍රියාත්මක කිරීමට ප්‍රථමයෙන් එහි අඩංගු ප්‍රකාශනයෙන් ප්‍රකාශනය යන්ත්‍ර භාෂා විධාන බවට පරිවර්තනය කර එසැණින් ම ඊට අනුරූපී විධාන ධාවනය කරමින් පරිවර්තන ක්‍රියාවලිය සිදු කරන භාෂා පරිවර්තකය අර්ථ වින්‍යාසකයක් ලෙස හැඳින්වෙයි.

අර්ථ වින්‍යාසකයක් භාවිත කරන පරිගණක භාෂාවල ක්‍රමලේඛ පරිවර්තනයේ දී

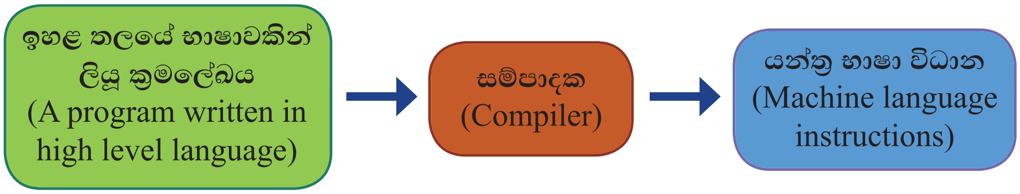
1. ක්‍රමලේඛයේ කාරක රීති දෝෂ නොමැති නම් ධාවනයෙන් පසු නිවැරදි ප්‍රතිදාන ලබා දෙයි.
2. ක්‍රමලේඛයේ කාරක රීති දෝෂ පවතී නම් එය අවසානය දක්වා ම ක්‍රියාත්මක නො වේ. (දෝෂය ඇති ස්ථානය දක්වා ක්‍රියාත්මක වීමට හැකියාව ඇත.)

වැදගත්

ක්‍රමලේඛය ධාවනය කරන වාරයක් පාසා යන්ත්‍ර කේතය බවට පරිවර්තනය සිදු වේ.

සම්පාදක (Compiler)

ඉහළ තලයේ පරිගණක භාෂාවකින් ලියන ලද ක්‍රමලේඛය ක්‍රියාත්මක කිරීමට ප්‍රථමයෙන් එය සම්පූර්ණයෙන් ම එක වර යන්ත්‍ර භාෂා උපදෙස් බවට පරිවර්තනය කිරීමෙන් පසු ධාවනය කිරීම සම්පාදකය මගින් සිදු වේ.

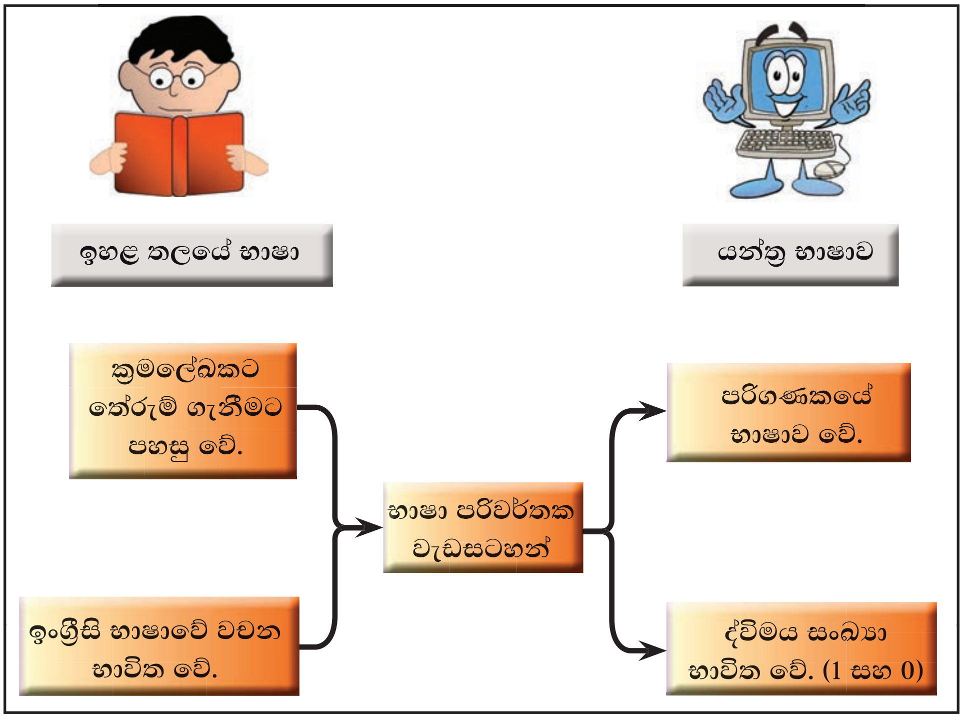


සම්පාදක භාවිත කරන පරිගණක භාෂාවල ප්‍රභව කේත (Source code) යන්ත්‍ර කේත බවට පරිවර්තනයේ දී

1. ක්‍රමලේඛයේ කාරක රීති දෝෂ නොමැති නම් ධාවනයෙන් පසු ප්‍රතිදාන ලබා දෙයි.
2. ක්‍රමලේඛයේ කාරක රීති දෝෂ පවතී නම් එය යන්ත්‍ර කේත බවට පරිවර්තනය නො වේ. එම දෝෂ සහිත ස්ථාන පෙන්වයි.

වැදගත්

ක්‍රමලේඛය එක් වරක් යන්ත්‍ර කේතය බවට පරිවර්තනය වූ පසු අවශ්‍ය ඕනෑ ම වාර ගණනක් ධාවනය කළ හැකි ය. නැවත පරිවර්තනය කිරීමක් අවශ්‍ය වන්නේ ප්‍රභව කේතය වෙනස් කළ හොත් පමණි.



සාරාංශය

- ගැටලුවක් විශ්ලේෂණය කිරීමෙන් ආදාන, ප්‍රතිදාන සහ ක්‍රියාවලිය හඳුනා ගත හැකි වේ.
- ඇල්ගොරිතම ගොඩනැගීමට ගැලීම් සටහන් සහ ව්‍යාජ කේත යොදා ගත හැකි වේ.
- ඕනෑ ම අනුක්‍රමයක් ලිවීමේ දී භාවිත වන මූලික අනුක්‍රමය, වරණය සහ පුනර්කරණය ලෙස පාලන වගුව ලෙස පවතී.
- ඇල්ගොරිතමයක පිහිටි පියවර කිහිපයක් හෝ සියල්ල ම හෝ ආරම්භක පියවරක සිට අවසාන පියවර දක්වා අනුපිළිවෙළින් ක්‍රියාත්මක වීම අනුක්‍රමය (Sequence) හැඳින්වේ.
- කොන්දේසියක් තෘප්ත වීම හෝ නොවීම හෝ අනුව ක්‍රියාත්මක කළ යුතු පියවර කුමක් ද යන්න තීරණය වන අවස්ථාවක් වරණයක් (Selection) ලෙස හැඳින්වේ.
- ඇල්ගොරිතමයක ඇතුළත් පියවරක් හෝ පියවර කිහිපයක් හෝ කොන්දේසියක් හෝ තෘප්ත වන තෙක් හෝ තෘප්ත ව පවතින තුරු හෝ නැවත නැවතත් සිදුවීම පුනර්කරණය (Repetition) ලෙස හැඳින්වේ.
- විචල්‍යයක්, නියතයක් හෝ ක්‍රමලේඛයක් හැඳින්වීම සඳහා යොදා ගන්නා නාමයක් හඳුන්වනයක් (identifier) ලෙස නම් කෙරේ.
- ක්‍රමලේඛය ක්‍රියාත්මක වන විට හඳුන්වන සඳහා පවරන ලද අගයන් වෙනස් වන හඳුන්වන, විචල්‍ය (Variable) ලෙස නම් කෙරේ.
- පැස්කල් යනු ඉහළ තලයේ ක්‍රමලේඛන භාෂාවකි.
- ප්‍රකාශනයක් සුළු වීම කාරකවල ප්‍රමුඛතා අනුපිළිවෙළ අනුව සිදු වේ.
- යන්ත්‍ර භාෂාව සහ ඇසෙම්බ්ලි භාෂාව පහළ තලයේ ක්‍රමලේඛන භාෂා ලෙස සැලකේ.
- PASCAL, BASIC, C හා JAVA වැනි භාෂා ඉහළ තලයේ ක්‍රමලේඛන භාෂා සඳහා උදාහරණ වේ.
- යන්ත්‍ර භාෂාවෙන් ලියන ලද ක්‍රමලේඛයක් ඍජු ව ම සකසනයට ධාවනය කළ හැකි වේ.
- යන්ත්‍ර භාෂාවෙන් හැර වෙනත් ඕනෑ ම පරිගණක භාෂාවකින් ලියන ලද ක්‍රමලේඛයක් ධාවනය කිරීමට ප්‍රථමයෙන් යන්ත්‍ර භාෂා උපදෙස් බවට පරිවර්තනය කළ යුතු ය.
- පටිපාටිගත, ප්‍රකාශන, ව්‍යුහගත, වස්තු නැඹුරු, ක්‍රමලේඛ හා පිටපත් ලේඛන ලෙස ක්‍රමලේඛ පැරඩයිම ඇත.
- අර්ථවින්‍යාසක (Interpreter) සහ සම්පාදක (Compiler) යනු භාෂා පරිවර්තක වැඩසටහන් දෙකකි.