පරිගණක ක්‍රමලේඛන සංකල්ප

1 පරිගණක ක්‍රමලේඛනයක් අවශය වන්නේ ඇයි ?

යමුනා :- අයියේ ඔයා දන්නවද ක්‍රමලේඛන භාෂා කියන්නේ මොනවාද කියලා. හෙට අපේ පාඩම ඒ ගැන. ඉතිං අපිට ඒ පාඩම කියවලා එන්න කිව්වා අපේ තොරතුරු තාක්‍ෂණය සර්.. ඒත් මට ටිකක් අමාරුයි පාඩම.

නිමල් :- ඉතිං නංගී ඔයා දන්නවානේ භාෂාවක් කියන්නේ මොකක් ද කියලා. මිනිසුන් මිනිසුන් අතර අදහස් හුවමාරු කරගන්න තමයි භාෂාවක් යොදා ගන්නේ.

යමුනා :- ක්‍රමලේඛන භාෂා කියන්නෙත් ඒ වගේ ම භාෂාවක් ද ?

නිමල් :- ඔව් නංගී ක්‍රමලේඛන භාෂා කියන්නෙත් ඒ හා සමාන දෙයක්. නමුත් මිනිසුන් භාවිත කරන භාෂාවලට වඩා තරමක් වෙනස්. ක්‍රමලේඛන භාෂා නමින් හඳුන් වන්නේ පරිගණකයක් සමග අදහස් හුවමාරු කරගැනීම සඳහා අවශ්‍ය වැඩ සටහන්, මෘදුකාංග සකසා ගන්නා භාෂාවලට. මේවා පරිගණක භාෂා ලෙසත් හඳුන්වනවා.

යමුනා :- ඉතින් ඇයි ක්‍රමලේඛන භාෂාවක් පරිගණකයට අවශ්‍ය වෙන්නේ ?

නිමල් :- ඇත්තට ම නංගී ක්‍රමලේඛන භාෂාවක් පරිගණකයේ තිබුනට වැඩක් නෑ. ඒවා අවශ්‍ය වෙන්නේ පරිගණක මෘදුකාංග නිර්මාණය කරන්න. පරිගණකයේ වැඩ කටයුතු ඉටු කරගන්නා මෘදුකාංග හදලා තියෙන්නේ ඔය කියන පරිගණක ක්‍රමලේඛන භාෂාවකින්. ක්‍රමලේඛන භාෂාවන් ගණනාවක් දැනට තියෙනවා. ඉන් එකක් තමයි ඔයාලට ඉගෙන ගන්න ලැබෙන පැස්කල් කියන පරිගණක ක්‍රමලේඛන භාෂාව.

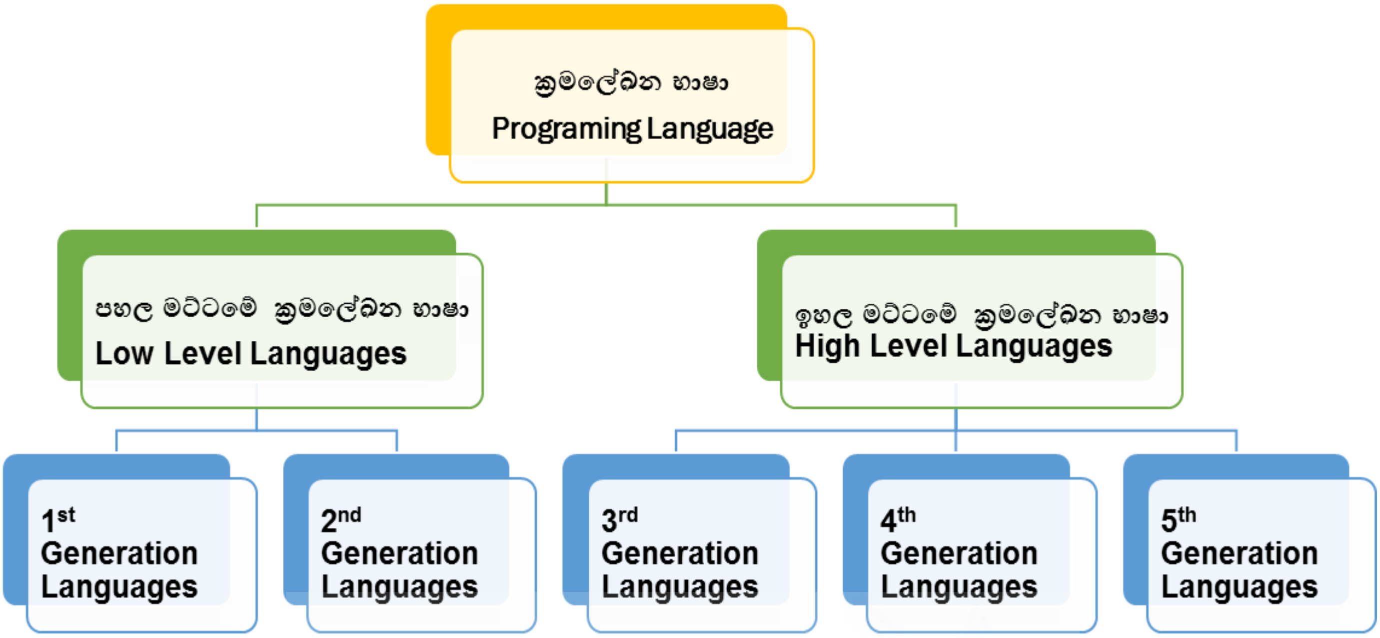
යමුනා :- හරි හරි අයියේ මට දැන් තේරුණා ක්‍රමලේඛන භාෂාවක වැදගත්කම මොකක්ද කියලා. අපිට හෙට පුලුවන් වේවි ඒවා ගැන තව දුරටත් ඉගෙන ගන්න. ගොඩාක් ස්තුතියි කියලා දුන්නාට.

2 ක්‍රමලේඛන භාෂා සහ එහි පරිණාමය

පරිගණක ක්‍රමලේඛනයක් යනුවෙන් අදහස් කරනුයේ පරිගණකයට තේරුම් ගත හැකි ආකාරයට ක්‍රමලේඛන භාෂාවක් භාවිත කරමින් දෙන ලද නිශ්චිත කාර්යයක් ඉටු කර ගැනීම සඳහා අනුපිළිවෙලින් සකස් කරන ලද උපදෙස් මාලාවකි. මෙසේ සකස් කරන ලද පරිගණක වැඩසටහන් පරිගණකය තුළ ස්ථාපනය කර ක්‍රියාත්මක කිරීමෙන් අපට අවශ්‍ය කරන කාර්යයන් ඉටු කර ගත හැකිය.

පරිගණක වැඩසටහන් සකසාගත හැකි පරිගණක ක්‍රමලේඛන භාෂා ගණනාවක් ඇත. මෙම භාෂා හඳුනාගැනීම සඳහා විවිධ වර්ගීකරණයන් ඉදිරිපත් කර තිබේ. හඳුනා ගැනීමේ පහසුව සඳහා ඉන් කිහිපයක් සලකා බලමු.

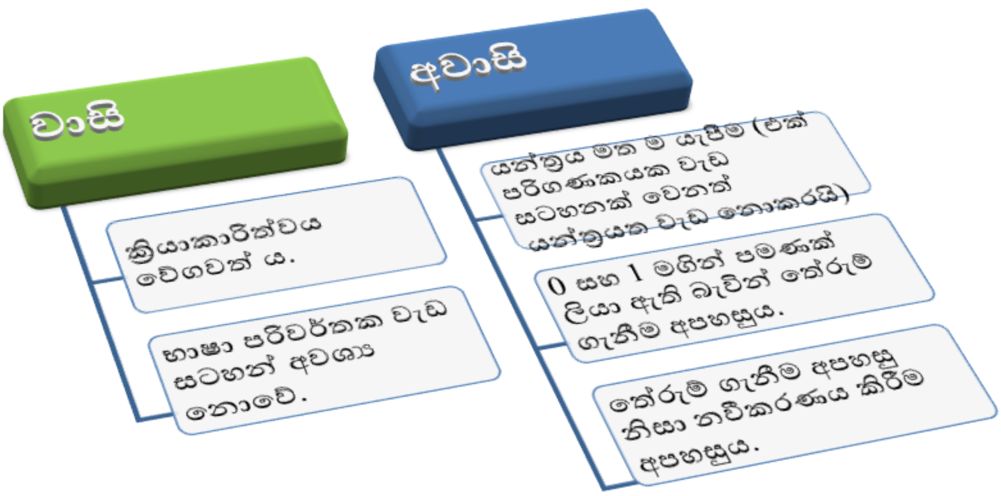
ක්‍රමලේඛන භාෂා වර්ගීකරණය



පහල මට්ටමේ ක්‍රමලේඛන භාෂා (Low Level Languages)

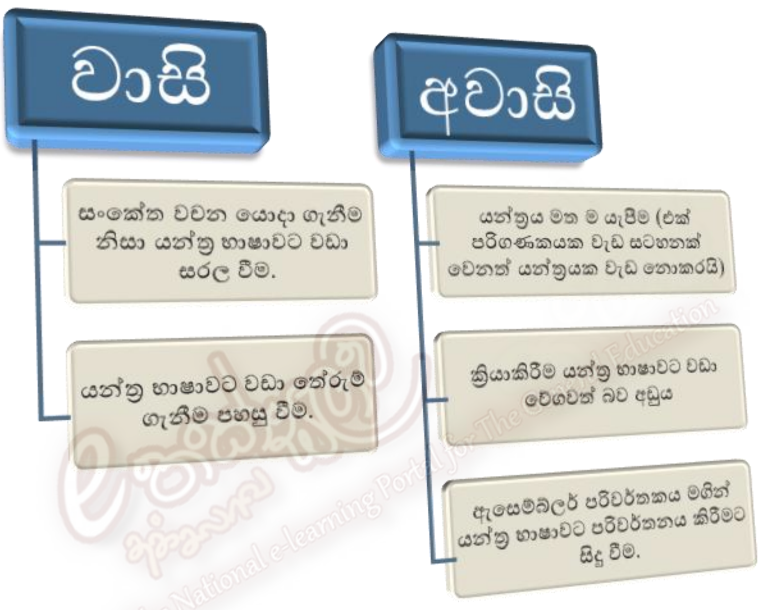
1 පලමු පරම්පරාවේ ක්‍රමලේඛන භාෂා (Machine Language)

- වැඩ සටහන් ලියන ලද්දේ ද්වීමය සංඛ්‍යා භාෂාව වන 0 හා 1 භාවිතා කර යන්ත්‍ර භාෂාවෙනි (Machine Code)
- විධාන ඇතුලත් කිරීම ඉතා සංකීර්ණ කාර්යයක් විය.
- යන්ත්‍රය මත යැපෙන භාෂාවකි.
- ක්‍රමලේඛකයා විසින් පරිගණකයේ දෘඩාංග **(Hardware)** පිලිබඳව දැනගත යුතුවේ.
- පරිවර්තක මෘදුකාංගයක් අවශ්‍ය නොවීය.
- වැඩසටහන් ක්‍රියාත්මක වීම වේගවත්ය. එහෙත් වැඩසටහන නවීකරණය කිරීම දුෂ්කරය.



2 දෙවන පරම්පරාවේ ක්‍රමලේඛන භාෂා (Assembly Language)

- විධාන ක්‍රමය **(Machine Language)** සාපේක්ෂව සරලය.
- විධාන ගනනාවක් වෙනුවට සංකේත භාවිතා කිරීම නිසා සංකේත භාෂා ලෙස හඳුන්වනු ලැබේ.
- කේත ලිවීම සඳහා ස්මෘති උපක්‍රම සහ විචල්‍යයන් (Mnemonics and variables) භාවිතා කරයි.
- **Assembler** නම්වූ සුවිශේෂී භාෂා පරිවර්තක මෘදුකාංගයක්, සංකේතාත්මක භාෂාව යාන්ත්‍රික භාෂාව බවට පරිවර්තනය කිරීම සඳහා යොදා ගනී.
- යන්ත්‍ර මත යැපෙන භාෂාවකි.
- පලමු පරම්පරාවට වඩා ක්‍රියාත්මක වන වේගය අඩුය.



ඉහල මට්ටමේ ක්‍රමලේඛන භාෂා - (High Level Language)

3 තුන්වන පරම්පරාවේ ක්‍රමලේඛන භාෂා (High Level Language - 3rd Generation languages)

- පරිගණක ක්‍රමලේඛනය කිරීමේ භාෂාව සංකීර්ණත්වය අඩුය. එයට හේතුව මෙහිදී භාවිතා වන වචන සහ සංකේත සරල හා පහසු වීමය.
- මිනිස් බසට බෙහෙවින්ම සමාන බැවින් අවබෝධ කර ගැනීම පහසුය.
- මේවාට පටිපාටිගත **(Procedural)** භාෂා යැයිද කිව හැක.
  උදා :- **FORTRAN, COBOL, PASCAL, C, C++**
- වැරදි හඳුනා ගැනීමේ පණිවිඩ දෙනු ලබන අතර ක්‍රම ලේඛය පරීක්ෂා කිරීම පහසුවේ.
- පරිගණක යන්ත්‍ර මත යැපෙන භාෂාවක් නොවේ.
- වැඩසටහනට අදාල කේත ලිවීම සහ නවීකරණය කිරීම පහසුය.
- ක්‍රම ලේඛකයකු පුහුණු කිරීම පහසුවේ.
- පරිවර්තක මෘදුකාංගයක් අවශ්‍ය වේ.
- යෙදුම් මෘදුකාංග සංවර්ධනය සඳහා භාවිත විය.

4 හතරවන පරම්පරාවේ ක්‍රමලේඛන භාෂා (High Level Language - 4thGeneration languages)

- වැඩ සටහනක් ලිවීමට ගතවන කාලය අඩු අතර ලිවීම සහ නවීකරණය කිරීම පහසුය.
- තෙවන පරම්පරාවේ භාෂාවකින් ලබාදෙන උපදෙස් කිහිපයක් සඳහා හතරවන පරම්පරාවේ භාෂාවේ එක් උපදෙසක් යොදා ගත හැකිය.
- පටිපාටිගත නොවන **(Non-procedural Language)** භාෂා වේ.**(Eg :- SQL)**
- මෘදුකාංග සංවර්ධනය සඳහා යන පිරිවැය අවමය.
- යන්ත්‍ර භාෂාව (**Machine Language**) හැරවීම සඳහා පරිවර්තන මෘදුකාංගයක් අවශ්‍ය වේ.
  උදා :- **Focus, Ideal, dbase III Plus, SQL**

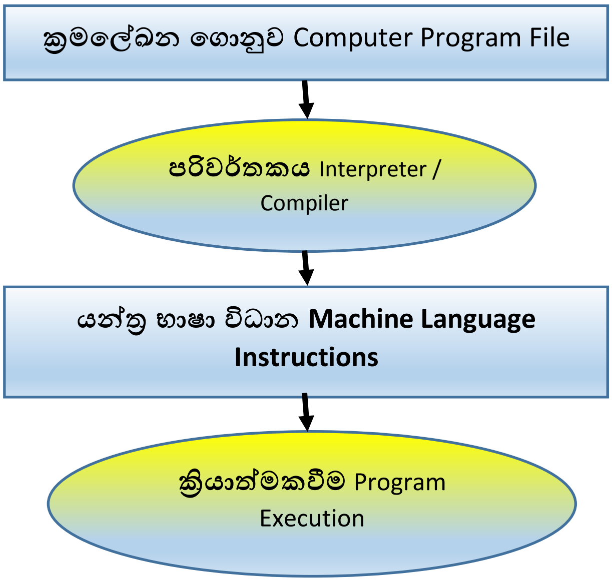
5 පස් වන පරම්පරාවේ ක්‍රමලේඛන භාෂා (Artificial Language - 5thGeneration languages)

- තීරණ ගැනීම සඳහා පරිගණක මාධ්‍ය මගින් නිර්මාණ කිරීමට භාවිත කරයි.
- පරිගණක වලට මිනිසුන් භාවිතා කරන භාෂා තේරුම් ගැනීමට පහසුවන මෘදුකාංග නිර්මාණය කිරීමට ද භාවිත වේ.
- වඩාත් බහුලව භාවිතා කරන භාෂා දෙකක් වන්නේ **LISP** සහ **Prolog** ය.
- රොබෝ යන්ත්‍ර වල දී උත්තේජක වල ප්‍රතිචාර දැක්වීම සඳහා වැඩසටහන් නිර්මාණය කිරීමටද භාවිත කරයි.

4 ක්‍රමලේඛයක් පරිගණකයක් තුල ක්‍රියාත්මක වන අකාරය

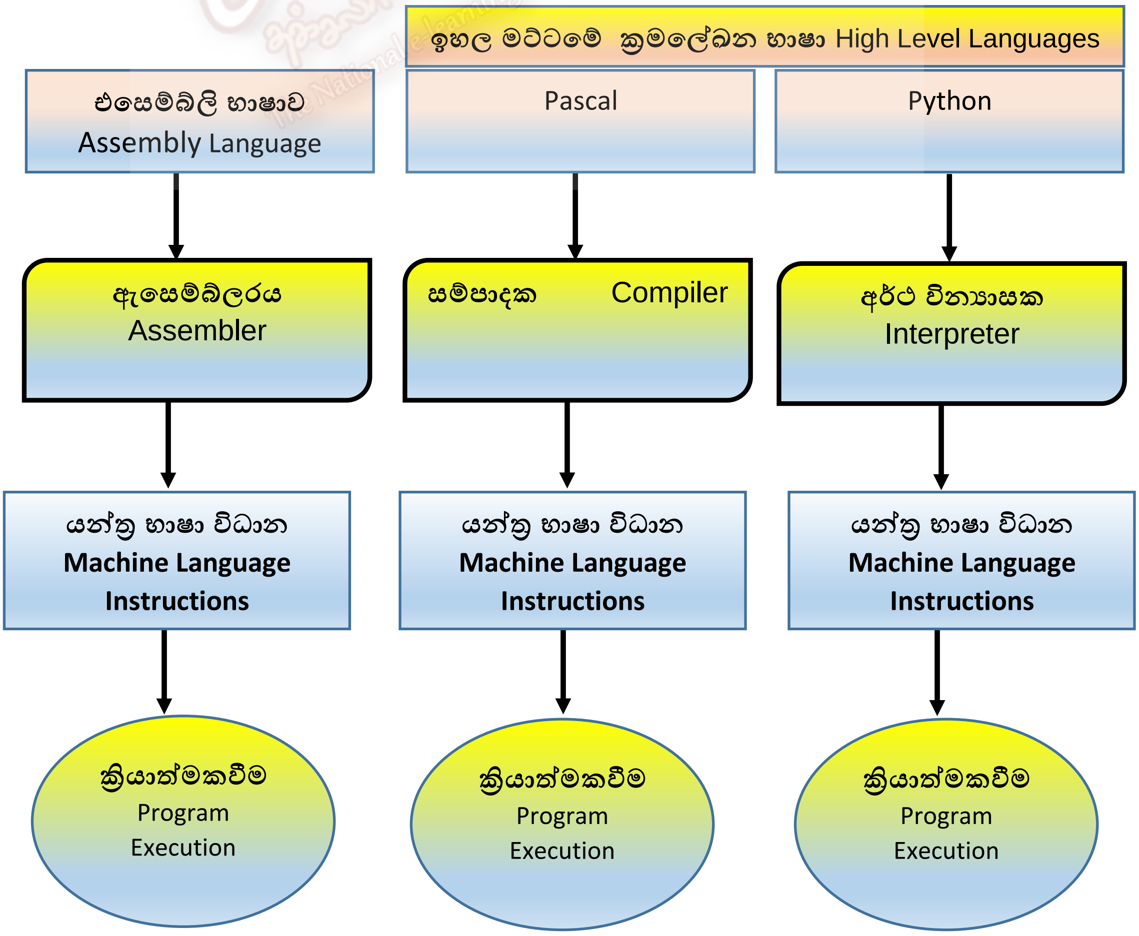
ක්‍රමලේඛනයක් පරිගණකය තුළ ක්‍රියාත්මකවන ආකාරය සලකන විට එහි අපට නොපෙනෙන ක්‍රියාවලියක් පවතී. බොහෝවිට අපි දකින්නේ පරිගණකයක වැඩ සටහනකට ආදානයන් සහ විධානයන් ලබා දුන් පසුව ප්‍රතිදානයක් ලැබීමයි. එහෙත් මෙම ක්‍රියාදාමය සංකීර්ණ වූවකි.

ක්‍රමලේඛනයක් පරිගණකය තුළ ස්ථාපිත කළ පසු එහි ක්‍රියාකාරිත්වය සිදුවනුයේ පරිගණකයට ලැබෙන විධානයන් අනුව අදාල ක්‍රමලේඛන ගොනුව භාෂා පරිවර්තන මෘදුකාංගයක් මගින් යන්ත්‍ර භාෂාවට හරවනු ලබයි. ඉන්පසුව අදාල ක්‍රමලේඛනය හඳුනාගෙන ඒ අනුව කටයුතු කිරීමට පටන් ගනී. පහත රූප සටහනෙන් දැක්වෙන්නේ එම පියවරයන් ය.



5 ක්‍රමලේඛන භාෂා පරිවර්තන මෘදුකාංග

යන්ත්‍ර භාෂාව හැර වෙනත් ඕනෑම පරිගණක ලේඛන භාෂාවකින් ලියන ලද පරිගණක වැඩ සටහනක් පරිගණකයේ ක්‍රියාත්මක කිරීම සඳහා ක්‍රමලේඛන භාෂා පරිවර්තන මෘදුකාංගයක් භාවිත කළ යුතුය. දැනට මෙලෙස භාවිතයේ පවතින ක්‍රමලේඛන භාෂා පරිවර්තන මෘදුකාංග තුනක් හඳුනා ගත හැකිය. එවා ඇසෙම්බ්ලරය (Assembler), සම්පාදක (Compiler) හා අර්ථ වින්‍යාසක (Interpreter) නම් වේ.



6 ගැටළුවක් විසඳීමේ පියවර

ගැටළුවක් යනුවෙන් අදහස් කරනුයේ අප අවට ලෝකයේ සිදුවන, සිදුවෙමින් පවතින හෝ සිදුවෙතැයි අපේක්ෂා කරන යම් සංසිද්ධියකි. නැතහොත් සිදු වීමකි. මෙම ගැටළුව පරිගණකයක් තුළින් විශ්ලේෂණය කර විසඳා ගැනීම සඳහා ක්‍රමලේඛන ගොඩ නැංවිය හැකිය. එහිදී ගැටළුව විසඳීම සඳහා යොදා ගන්නා වූ අමුද්‍රව්‍ය හෝ විධාන “ ආදාන ” (Input) ලෙස ද එම ගැටළුව විසඳීමෙන් ලබා ගන්නා වූ නිගමනය “ ප්‍රතිදාන ” (Output) ලෙස ද හඳුන්වනු ලබයි. මෙහිදී පරිගණකය ගැටළුව විසඳීම සඳහා ගනු ලබන ක්‍රියාවලිය “ සැකසීම ” (Process) ලෙස ද හඳුන් වනු ලබයි.

මූලික අවස්ථා කිහිපයක් පවතී

- ගැටළුව තේරුම් ගැනීම සහ විශ්ලේෂණය කිරීම **(Understanding the problem)**
- ගැටළුව සඳහා ඇල්ගොරිතමය (**Algorithm**) ගොඩනැගීම **(Developing an Algorithm for the problem)**
- පරිගණක ක්‍රමලේඛය ලිවීම **(Writing a Computer Program)**
- පරිගණක ක්‍රමලේඛය ක්‍රියාක්මකකර බැලීම **(Testing the Computer program)**

ගැටළුව තේරුම් ගැනීම සහ විශ්ලේෂණය

උදාහරණයක් ලෙස ඔබගේ කාමරයේ වර්ග ඵලය කොපමණදැයි දැනගැනීමට ඔබට අවශ්‍ය වෙයි. එහිදී ඔබගේ ගැටළුව වනුයේ කාමරයේ වර්ග ඵලය කීයද යන්නයි. ඔබ එම ගැටළුව විසඳීම සඳහා කාමරයේ දිග x පළල මගින් වර්ගඵලය සෙවිය හැකි යැයි දැන ගනියි. එවිට ඔබ කරනුයේ කාමරයේ දිග සහ පළල මැනීමයි. අපි ඉහත සඳහන් කළ අමු ද්‍රව්‍ය නැතහොත් ආදානය වන්නේ කාමරයේ දිග සහ පළලයි. සැකසීම වනුයේ දිග සහ පළල ප්‍රමාණයන් ගුණ කිරීමේ ක්‍රියාවලියයි. ප්‍රතිදානය වනුයේ වර්ගඵලයයි.

ගැටළුව විශ්ලේෂණය කිරීමේදී ආදාල ප්‍රතිදාන සහ ක්‍රියාවලිය ගැන අවබෝධ කර ගත යුතුය.

ආදාන : කාමරයේ දිග සහ පළල
ක්‍රියාවලිය : කාමරයේ වර්ගඵලය = දිග x පළල
ප්‍රතිදාන : කාමරයේ වර්ගඵලය

ඇල්ගොරිතමය නිර්මාණය (ගැලීම් සටහන් හා ව්‍යාජ කේත)

තාත්තා :- පුතා! අවිස්සාවේල්ලෙ මාමල ආවා. අම්මත් ගෙදර නැනේ, ඔයාට පුළුවන්ද තේ එකක් හදල දෙන්න.
පුතා :- අනේ තාත්තේ මම දන්නේ නැනේ.
නැන්දා :- හරි පුතා, අපි තේ හදාගමු. එන්න මම කියා දෙන්නම්.

මුලින්ම 1. වතුර උණු කරගන්න.
2. තේ කොළ පෙරනයට දාන්න.
3. පෙරනය තුලින් උණු ජලය තේ කෝප්පයට දාන්න.
4. සීනි තේ හැදි 2ක් දමා දිය කරන්න.
5. දැන් තේ කෝප්පය පිළිගන්වන්න.

තාත්තා :- ඇහුනද පුතා.නැන්දා කිව්ව දේ, ඒක ඇල්ගොරිතමයක්.
පුතා :- මොකක්ද තාත්තේ ඇල්ගෝරිතමයක් කියන්නේ?
තාත්තා :- මේ පොත කියවල බලන්නකෝ.

බුද්ධිමත් බව මිනිසුන් අනෙක් සතුන්ගෙන් වෙනස්වීමට ප්‍රධාන හේතුවක් වන්නේය. දෛනික ජීවිතයේදී හමුවන සියළු ගැටළු හෝ අවස්ථා සඳහා තීරණ ගැනීම අවස්ථානුකූලව මිනිසා විසින් සිදු කරනු ලබයි. බුද්ධිය හෝ අවබෝධය භාවිතා නොකර , ඔබට එක් දිනක් ජීවත් විය හැකි ද? පිළිතුර සත්තකින්ම බැහැ යන්නයි. කඩයකින් බඩුවක් මිලදී ගැනීමේදීත්, බැංකුවකින් මුදලක් ලබාගැනීමේදීත්, වෙනත් ඕනෑම අවස්ථාවකදීත්, ඔබට තීරණ ගැනීමට සිදුවේ. මේවා සියල්ල ඇල්ගොරිතම සඳහා උදාහරණ වේ.

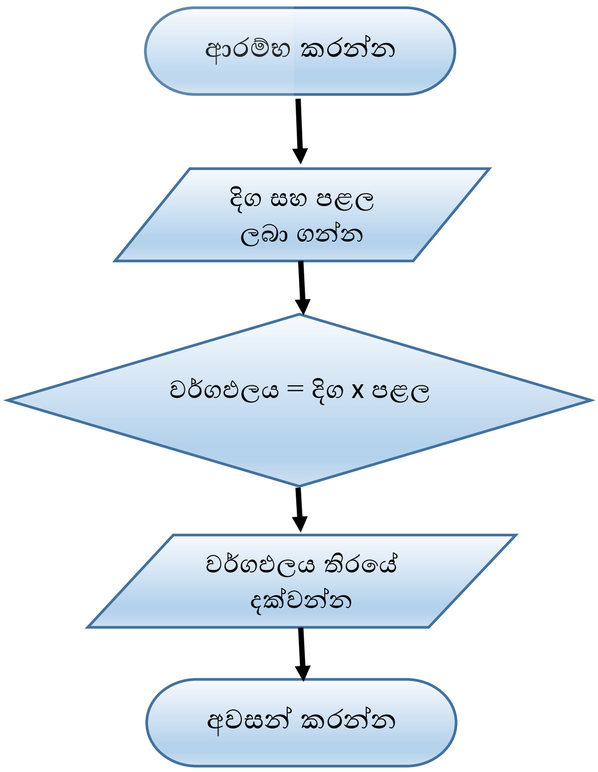
අවස්ථාව 1. :- යමෙක් ඔබෙන් වෙලාව ගැන විමසුවොත්? ඔබ ගන්නා පියවර ගැන සිතන්න.
අවස්ථාව 2. :- සඳුදා දිනයේ ඔබ පාසලට ගෙන යා යුතු පොත් ටික මෙතෙක් තෝරා ගෙන නැත්නම්, සඳුදා උදයට ඔබ මොකද කරන්නේ? ගන්නා පියවර ගැන සිතන්න.

ඇල්ගොරිතමයක් යනු, යම් බලාපොරොත්තු වන අරමුණක් කරා ලඟා වීමට ගත යුතු පියවරයන් වේ. ඉතා විවෘත ලෙස සැලකීමේදී, ගැටළුවක් යනු යම් ප්‍රතිඵලයකට ලඟා වීමට ඇති බාධාවකි. ගැටළු විසඳීම (Problem solving) යනු එම බාධකය ඉවත් කිරීම සඳහා ගනු ලබන ක්‍රියාමාර්ග සමුච්චයයි. කෙසේ වෙතත් ලොව ඇති සියළු ගැටළු විසඳිය නොහැක. ඒවා විවෘත ගැටළු වන අතර විසඳිය හැකි සෑම ගැටළුවක් සඳහාම ඇල්ගොරිතම ලිවිය හැක. එය පියවර අනුක්‍රමයකි. අද පරිගණක වැඩසටහන් ලිවීමේ දී ද භාවිත කරයි. යම් කාර්යක් පරිගණකය හරහා සිදුකර ගැනීමේ දී, පියවරින් පියවර අදාල උපදෙස් ඊට සැපයිය යුතු වන්නේය. ඒ අනුව පරිගණක වැඩසටහනක් යනු දීර්ඝ වශයෙන් පියවරින් පියවර ලියන ලද උපදෙස් මාලාවක් වන අතර ඊට බාහිරින් ලැබෙන දේ ආදාන (Input) ලෙසත්, ප්‍රථිඵලය ප්‍රතිදාන (Output) ලෙසත් හඳුන්වනු ලැබේ.

ඇල්ගොරිතමයක් යනුවෙන් අදහස් කරනුයේ යම් ගැටළුවක් විසඳීම සඳහා අනුගමනය කරනු ලබන පියවර සමූහයටයි. අදාල ගැටළුව තේරුම් ගෙන සැලසුමක් මත සැකසූ පියවරයන් අනුව අවසාන නිගමනයක් කරා එළඹිය හැකි නම් එම ඇල්ගොරිතමය ඵලදායක වන්නේය.

<u>ඉහත දැක්වූ උදාහරණයට අදාල කාමරයේ වර්ග ඵලය සෙවීම සඳහා ඇල්ගොරිතමයට අදාල ගැලීම් සටහනක් නිර්මාණය</u>

ඇල්ගොරිතමය



ක්‍රමලේඛය ලියා දැක්වීම

ඇල්ගොරිතමය, ක්‍රමලේඛන :- (පරිගණක) භාෂාවක් ආධාරයෙන් ලියා දැක්වීම මෙහිදී සිදු කරයි. මෙය කේතනය (**Coding**) යනුවෙන් හඳුන්වනු ලබයි.

ක්‍රමලේඛය ක්‍රියාත්මක කිරීම

ලියන ලද ක්‍රමලේඛය ක්‍රියාත්මක කර නිවැරදි දැයි මෙහිදී පරීක්ෂා කරනු ලැබේ. වැරදි තිබේ නම් එම කොටස් ක්‍රමලේඛක විසින් (**Programmer**) නිවැරදි කරනු ලැබේ. මෙය දෝෂ හරණය හෙවත් (**debugging**) ලෙස හඳුන්වයි.

7. ඇල්ගොරිතම නිර්මාණයට අදාල පාලන ව්‍යුහ

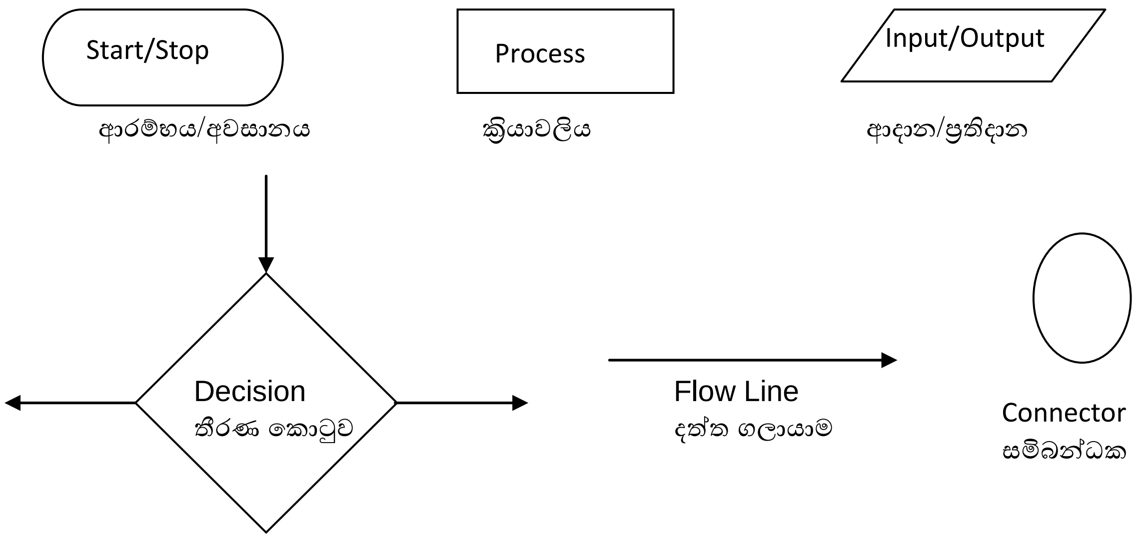
ඇල්ගොරිතම (Algorithm) නිරූපණය සඳහා භාවිත කළ හැකි විවිධ මෙවලම් (tools) කිහිපයක් පවතී.

1 ගැලීම් සටහන් ක්‍රමය (Flow Chart)
2 ව්‍යාජ කේත ක්‍රමය (Pseudo Code)

❖ ගැලීම් සටහන් ක්‍රමය (Flow Chart)

ක්‍රමලේඛයක ක්‍රියාකාරිත්වය අනුපිළිවෙළින් දැක්වෙන රූපමය නිරූපණය ගැලීම් සටහනක් ලෙස හැඳින්වේ. ගැලීම් සටහන දත්ත ගලායන ආකාරයත් ඒවා යම් යම් ක්‍රියාකාරකම් වලට භාජනය වන ආකාරයත් පෙන්නුම් කරයි.

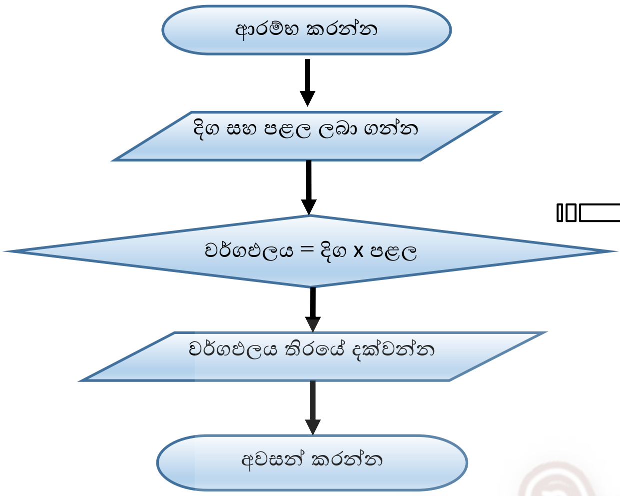
ගැලීම් සටහන් ඇඳීම සඳහා භාවිතා කරන සංකේත.



❖ ව්‍යාජ කේත ක්‍රමය

ගැටළුවක් විසඳීම සඳහා අනුගමනය කරන ක්‍රමවේදය සංක්ෂිප්තව මානව භාෂාවක් (Human Language) මගින් ඉදිරිපත් කිරීම ව්‍යාජ කේතයක් (Pseudo Code) ලෙස හැඳින්වේ. මෙය පරිගණක ක්‍රමලේඛකයකු නොවන කෙනෙකුට වුවද තේරුම් ගත හැකි ආකාරයට සරල බසින් ලියා දක්වනු ලැබේ.

උදා 1 :- කාමරයක වර්ග ඵලය සෙවීම



ගැලීම් සටහන

ව්‍යාජ කේතය

උදා 2 :-

අරය (**r**) ඇති වෘත්තයක වර්ගඵලය සොයන්න.

ඇතුලත් කළ යුතු අගය (Input)	=	අරය
බලාපොරොත්තු වන ප්‍රතිඵලය (Output)	=	වෘත්තයේ වර්ගඵලය.

ඇල්ගෝරිතමය - (ව්‍යාජ කේතයක් ලෙස)

පියවර 1.	ආරම්භ කරන්න.
පියවර 2.	**r** අගය ලබා ගන්න.
පියවර 3.	වර්ගඵලය = PI × **r** × **r**
පියවර 4.	වර්ගඵලය තිරය මත පෙන්වන්න.
පියවර 5.	අවසන් කරන්න.

උදා 3 :-

සංඛ්‍යා 2 ක එකතුව සොයන්න.

ලබා ගත යුතු අගයන්	=	පළමු සංඛ්‍යාව හා දෙවන සංඛ්‍යාව
බලාපොරොත්තු වන ප්‍රතිඵලය	=	සංඛ්‍යා දෙකෙහි එකතුව.

ඇල්ගොරිතමය :-

පියවර 1	:- ආරම්භ කරන්න.
පියවර 2	:- පළමු සංඛ්‍යාව ලබා ගන්න.
පියවර 3	:- දෙවන සංඛ්‍යාව ලබා ගන්න.
පියවර 4	:- මුළු අගය = පළමු අගය + දෙවන අගය.
පියවර 5	:- සංඛ්‍යාව මුද්‍රණය කරන්න.
පියවර 6	:- අවසන් කරන්න.

ඇල්ගොරිතම ප්‍රධාන වශයෙන් ආකාර 3 කින් පවතින බව නිරීක්ෂණය කළ හැකිය. මේවා පාලන ව්‍යුහ හෙවත් (Control Structures) ලෙස හැඳින්වේ.

1. අනුක්‍රමය.-Sequence
2. වරණය.-Selection
3. පුනර්කරණය.-Repetition

8 ඇල්ගොරිතමය පරිගණක ක්‍රමලේඛයක් බවට පරිවර්තනය කිරීම.

මේ සඳහා විෂය නිර්දේශයේ සඳහන් පැස්කල් ක්‍රමලේඛන භාෂාව භාවිත කර ඇත.

❖ පැස්කල් ක්‍රමලේඛන භාෂාව භාවිතය

➢ හඳුන්වන.- (Identifiers)

හඳුන්වනයක් යනු පරිගණකයේ මතකය තුල දත්ත තැම්පත් කිරීමට යොදා ගන්නා අවකාශයක් සඳහා භාවිත කරන නමකි. මෙම හඳුන්වනයක නාමය සැකසීම්දී පහත දැක්වෙන සාමාන්‍ය නීති අනුගමනය කල යුතුය.

- ඉංග්‍රීසි හෝඩියේ අක්ෂරයකින් ආරම්භ කල යුතුය.
- හඳුන්වනයේ පළමු අක්ෂරයට පසුව අක්ෂර - (a-z, A-Z) හෝ සංඛ්‍යාංක - (0-9) සහ යටිඉර (_)යොදා ගත හැකිය.
- වචන අතර පරතරයක් නොපැවතිය යුතුය.
- හඳුන්වනයක් තුල පහත සඳහන් අනුලක්ෂණ අඩංගු නොවිය යුතුය.
  ~ ! @ # $ % ^ & * ( ) - + = { } [ ] : ; ' " <> ? , . / | \

හඳුන්වන සඳහා අර්ථවත් නාමයන් යොදා ගැනීමෙන් ක්‍රමලේඛන තේරුම් ගැනීම පහසු කරයි.

○ **වලංගු හඳුන්වන සඳහා උදාහරණ**

FirstName , Middle_Name, Last_Name, Sum, SUM, Total_Nos , Num1,

○ **වලංගු නොවන හඳුන්වනයන් සඳහා උදාහරණ**

9ABC, $125, Average Marks, First Name

➢ ක්‍රමලේඛන භාෂාවල භාවිත කරන සම්මත දත්ත ප්‍රරූප

Integer -	ධන හෝ සෘණ නිඛිල සංඛ්‍යා (පූර්ණ සංඛ්‍යා)
Real -	ධන හෝ සෘණ තාත්වික සංඛ්‍යා (දශම සංඛ්‍යා/විද්‍යාත්මක අංකනය සහිත අගයන්)
Boolean–	True (සත්‍ය) හෝ False (අසත්‍ය)
Char -	ඕනෑම අනුලක්ෂණයක්
String -	අනුලක්ෂණ වල එකතුවක්

○ දත්ත ප්‍රරූප සඳහා උදාහරණ

Integer	–	0,	84 ,	-12
Real	–	0.0 ,	25.68 , 2.40E+13 ,	6.52E-18
Boolean	–	True	හෝ	False
Char	–	'a' ,	'#' ,	' 7 '
String	–	'ICT' ,	'programming'	

සටහන : Char සහ String යන දත්ත ප්‍රරූප සඳහා යොදා ගන්නා අගයයන් (Values) ' ' උඩු කොමා (single quotation) තුල දක්වයි.

➢ ක්‍රමලේඛනයේදී යොදා ගන්නා විචල්‍ය සහ නියත

○ විචල්‍ය - (Variables)

ක්‍රමලේඛය ක්‍රියාත්මක වන විට හඳුන්වනයන් සඳහා පවරන ලද අගයන් වෙනස් වන හඳුන්වන, **විචල්‍යය** ලෙස නම් කරයි. පැස්කල් භාෂාවේදී විචල්‍ය අර්ථ දැක්වීම සඳහා "var" යන වචනය යොදා ගනී.

උදා:-

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

○ නියත - (Constant)

ක්‍රමලේඛය ක්‍රියාත්මක වන විට හඳුන්වනය සඳහා පවරන ලද අගයන් වෙනස් නොවන හඳුන්වන, **නියත** ලෙස නම් කරයි.පැස්කල් භාෂාවේදී නියත අර්ථ දැක්වීම සඳහා "const" යන වචනය යොදා ගනී.

උදා:-

Const Avogadro _Number = $6.023*10^{23}$;

Const pie = 22/7;

❖ කාරක - (Operators)

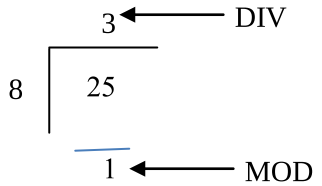
➢ කාරකවල අවශ්‍යතාවය

කාරක නොමැතිව ගණනය කිරීම , සන්සන්දනය කිරීම සහ තාර්කික ප්‍රකාශ ගොඩ නැගීම කල නොහැකිය. එබැවින් ක්‍රමලේඛයන් ලිවීමට කාරක අත්‍යාවශ්‍ය වේ.

➢ මූලික කාරක වර්ග

○ 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	20DIV 6	3
MOD	බෙදීමෙන් පසු ශේෂය	20 MOD 6	2



○ සැසඳුම් කාරක

අගයන් හෝ ප්‍රකාශ සන්සන්දනය කිරීම සඳහා සැසඳුම් කාරක යෙදා ගනී. සැසඳුම් කාරකයක් අඩංගු ප්‍රකාශයක අවසන් ප්‍රතිඵලය සෑම විටම බූලියානු අගයක් ගනී. එනම් ප්‍රකාශය සත්‍ය හෝ අසත්‍ය වේ.

කාරකය	භාවිතය	ප්‍රකාශය	ප්‍රතිඵලය
>	විශාල	9>3	සත්‍ය
>=	විශාල හෝ සමාන	7>=7	සත්‍ය
<	කුඩා	5<2	අසත්‍ය
<=	කුඩා හෝ සමාන	3<=9	සත්‍ය
=	සමාන	5=1	අසත්‍ය
<>	අසමාන	3<>3	සත්‍ය

○ තාර්කික කාරක

- **AND කාරකය**

ප්‍රකාශන දෙකක් සංසන්දනය කිරීම සඳහා මෙම මෙහෙයුම භාවිත කරනු ලබයි. “පළමු ප්‍රකාශය AND දෙවන ප්‍රකාශය” ලෙස මූලිකව යොදනු ලබයි. එහිදී පළමු හා දෙවන ප්‍රකාශ සත්‍ය හෝ අසත්‍ය වීම අනුව AND කාරකය ගැනීමෙන් ලැබෙන ප්‍රතිඵලයද සත්‍ය හෝ අසත්‍ය වේ.පහත වගුව මඟින් AND කාරකයේ ක්‍රියාකාරීත්වය දැක්වේ.

පළමු ප්‍රකාශය	දෙවන ප්‍රකාශය	පළමු ප්‍රකාශය AND දෙවන ප්‍රකාශය
අසත්‍ය	අසත්‍ය	අසත්‍ය
අසත්‍ය	සත්‍ය	අසත්‍ය
සත්‍ය	අසත්‍ය	අසත්‍ය
සත්‍ය	සත්‍ය	සත්‍ය

උදා:-

(3>=2) AND (3<>3)
3>=2 සත්‍ය වේ. 3<>3 අසත්‍ය වේ.
එබැවින් සමස්ත ප්‍රකාශය ද අසත්‍ය වේ.

- ❖ ප්‍රකාශ දෙකෙන් අවම වශයෙන් එක් ප්‍රකාශයක් හෝ අසත්‍ය වූවිට AND කාරකය සහිත ප්‍රකාශයද අසත්‍ය වේ.
- ❖ ප්‍රකාශ දෙකම සත්‍ය වනවිට පමණක් AND කාරකය සහිත ප්‍රකාශය සත්‍ය වේ.

- **OR කාරකය**

"පළමු ප්‍රකාශය OR දෙවන ප්‍රකාශය" ලෙස මූලිකව යොදනු ලබයි. එහිදී පළමු හා දෙවන ප්‍රකාශ සත්‍ය හෝ අසත්‍ය වීම අනුව OR කාරකය ගැනීමෙන් ලැබෙන ප්‍රතිඵලයද සත්‍ය හෝ අසත්‍ය වේ.පහත වගුව මඟින් OR කාරකයේ ක්‍රියාකාරීත්වය දැක්වේ.

පළමු ප්‍රකාශය	දෙවන ප්‍රකාශය	පළමු ප්‍රකාශය OR දෙවන ප්‍රකාශය
අසත්‍ය	අසත්‍ය	අසත්‍ය
අසත්‍ය	සත්‍ය	සත්‍ය
සත්‍ය	අසත්‍ය	සත්‍ය
සත්‍ය	සත්‍ය	සත්‍ය

උදා:-

(3>=2) OR (3<>3)
3>=2 සත්‍ය වේ. 3<>3 අසත්‍ය වේ.
එබැවින් සමස්ත ප්‍රකාශය සත්‍ය වේ.

- ❖ ප්‍රකාශ දෙකෙන් අවම වශයෙන් එක් ප්‍රකාශයක් හෝ සත්‍ය වූවිට OR කාරකය සහිත ප්‍රකාශයද සත්‍ය වේ.
- ❖ ප්‍රකාශ දෙකම අසත්‍ය වනවිට පමණක් OR කාරකය සහිත ප්‍රකාශය අසත්‍ය වේ.

- **NOT කාරකය**

සෑම විටම සත්‍ය ප්‍රකාශයක් NOT කාරකය මඟින් අසත්‍ය ලෙස දක්වන අතර අසත්‍ය ප්‍රකාශයක් NOT කාරකය මඟින් සත්‍ය ලෙස දක්වයි.

ප්‍රකාශය	NOT (ප්‍රකාශය)
අසත්‍ය	සත්‍ය
සත්‍ය	අසත්‍ය

උදා:- NOT(5=5) අසත්‍ය ලෙස දක්වයි.

➢ කාරක ප්‍රමුඛතාවය

ක්‍රමලේඛන භාෂා ක්‍රියාත්මක වීමේදී කාරකවල ප්‍රමුඛතා අනුපිළිවෙලක් පවතී.

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

උදා: 1. 5 + 15 MOD 4
= 5 + 3
= 8

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.

```
program add_Num(input,output);                      ← ක්‍රමලේඛයේ නම
var     number1: integer;
        number2: integer;                            විචල්‍ය හඳුන්වාදීම
        total : integer;
        avg: real;
Begin                                                ← ප්‍රධාන කොටසේ ආරම්භය
        Write('Welcome Pascal Programming');
        (* calculate total and  average of two numbers*) → විවරණ දැක්වීම
        Writeln('Enter First Number = ');
        Read(number1);                               ආදාන
ප්‍රතිදාන Writeln('Enter Second Number = ');
        Read(number2);
        total: =number1 + number2;
        avg := total/2;                              ක්‍රියාවලිය දැක්වෙන ප්‍රකාශ
        writeln('Total is = ' , total);
        writeln('Average is = ', avg);
End.                                                 ← ප්‍රධාන කොටසේ ආවසානය
```


- මෙහි “program” යනු ඇවිරුණු පදයකි.
- “input” සහ ”output” යනු මූල පදවේ.
- “add_Num” යනු හඳුන්වනයකි. මෙය ක්‍රමලේඛයේ නාමයයි.
- ක්‍රමලේඛයේ නාමය සමඟ වරහන් තුල input,output ලෙස දැක්වීම අත්‍යාවශ්‍ය නොවේ.
- ආදානය සඳහා read() සහ readln() ශ්‍රිත භාවිත කරයි.
- Read(number1); ප්‍රකාශය මඟින් number1 වූ විචල්‍ය සඳහා දත්ත ආදානය කරගනී.
- readln() මඟින් දත්ත ආදානය කර ගන්නේ නව පේළියක සිටය.
- ප්‍රතිදානය සඳහා write() සහ writeln() ශ්‍රිත භාවිත කරයි.
- Write(‘Welcome Pascal Programming’); ප්‍රකාශය මඟින් Welcome Pascal Programming නම් වූ පාඨ කොටස ප්‍රතිදානය කරයි.
- writeln(‘Average is = ’ , avg); ප්‍රකාශය මඟින් නව පේළියක Average is = යන පාඨ කොටස සමඟ avg විචල්‍යයේ අගය ප්‍රතිදානය කරයි.

➢ ක්‍රමලේඛයේ ක්‍රියාවලිය දැක්වෙන ප්‍රකාශ:-

- ප්‍රකාශයක කෙලවරේ semi-colon (;) යොදනු ලැබීම මඟින් එම ප්‍රකාශය අවසන් බව කියවේ.
- total: =number1 + number2; යන ප්‍රකාශයෙන් කියවෙන්නේ number1 සහ number2 විචල්‍ය දෙකේ අගයන් එකතු කර total නම් වූ විචල්‍යයට පවරනු ලබන බවයි.

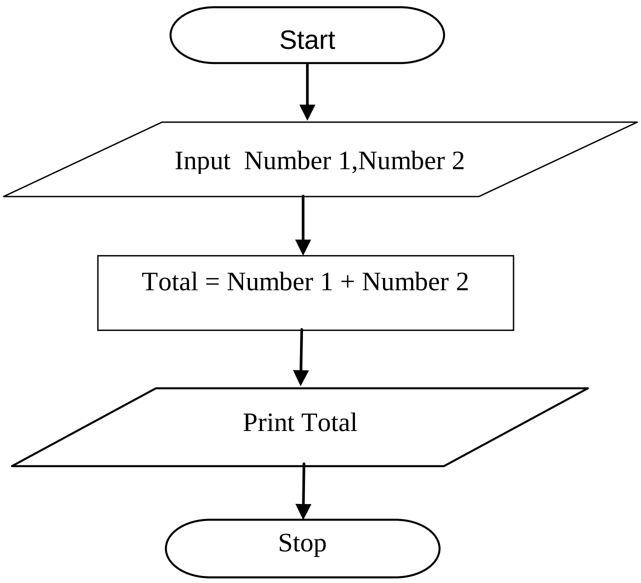
- “: =” Assignment Operator ලෙස හඳුන්වයි.

9 ඇල්ගොරිතම සංවර්ධනය කිරීම සඳහා පාලන ව්‍යුහ භාවිතය

➢ අනුක්‍රමය පාලන ව්‍යුහය සහිත ක්‍රමලේඛ

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

උදා:- 1. සංඛ්‍යා දෙකක් එකතු කිරීම



ගැලීම් සටහන

```
begin
        enter first_number
        enter second_number
        Total=first_number+second_number
        display Total
end.
```

ව්‍යාජ කේතය

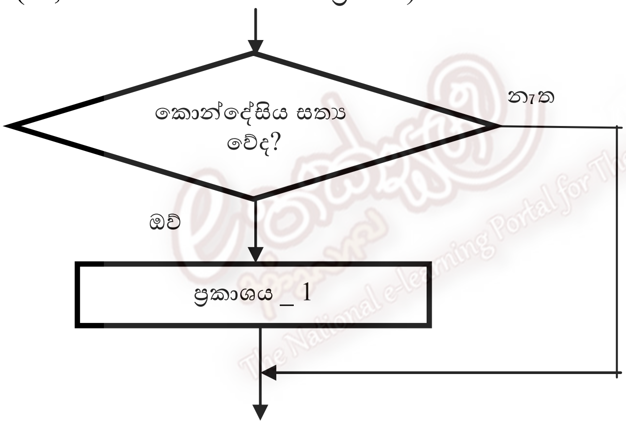
```
program Add_Number(input/output);
var    Number_1:Integer;
       Number_2:Integer;
       Total:Integer;
begin
        writeln('Enter first number');
        readln(Number_1);
        writeln('Enter second number');
        readln(Number_2);
        Total:= Number_1  + Number_2;
        writeln('Answer is ', Total);
        readln;
end.
```

පැස්කල් ක්‍රමලේඛය

➢ තේරීම් පාලන ව්‍යුහය (වරණය)-Selection සහිත ක්‍රමලේඛ

ඇල්ගොරිතමයක ඇතුළත් කොන්දේසියක් තෘප්ත වීම හෝ නොවීම හෝ අනුව ක්‍රියාත්මක කළ යුතු පියවර කුමක් ද යන්න තීරණය වන අවස්ථාවක් වරණයක් ලෙස හැඳින්වේ. වරණය සහිත ක්‍රමලේඛයක් තේරීම් පාලන ව්‍යුහය සහිත ක්‍රමලේඛයක් ලෙස හැඳින්වේ.

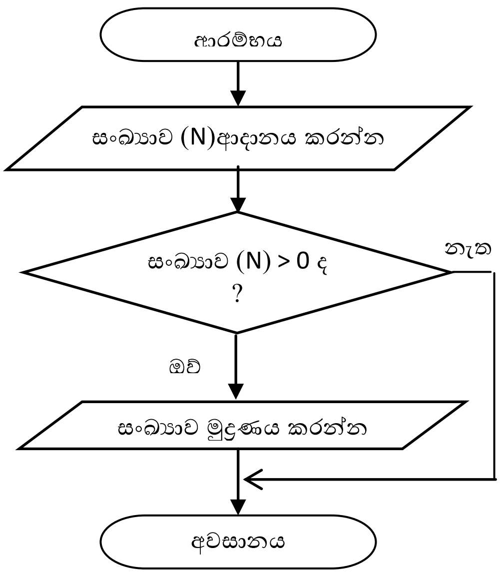
➢ **කොන්දේසිය තෘප්ත වන්නේ නම් පමණක් ප්‍රකාශය ක්‍රියාත්මක වී ඊට අදාල ප්‍රතිඵලය ලබා දෙයි.** (IF, THEN සහ END IF ප්‍රකාශ)




```
IF  < Condition >  THEN
        Statement_1
END IF
```


මෙහිදී කොන්දේසිය තෘප්ත වන්නේ නම් පමණක් ප්‍රකාශය _1 ට අදාල ප්‍රතිඵලය ලබා දෙයි.

උදා 2. ආදානය කරන සංඛ්‍යාව ධන සංඛ්‍යාවක් නම් පමණක් එම සංඛ්‍යාව මුද්‍රණය කිරීම.




```
Begin
        Input Number
        IF  Number>0  THEN
                Print Number
        END IF
End.
```

ව්‍යාජ කේතය

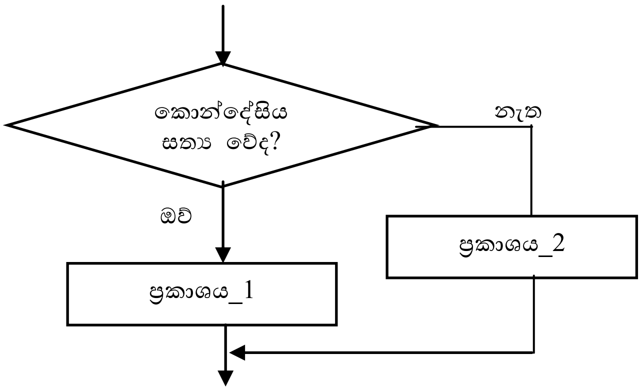
```
Program positive_No(input,output);
Var  Number : integer;
Begin
        Writeln('Enter Number = ');
        Read(Number);
        If  Number > 0  then
                Writeln('Positive Number');
        readln;
End.
```

පැස්කල් ක්‍රමලේඛය

ගැලීම් සටහන

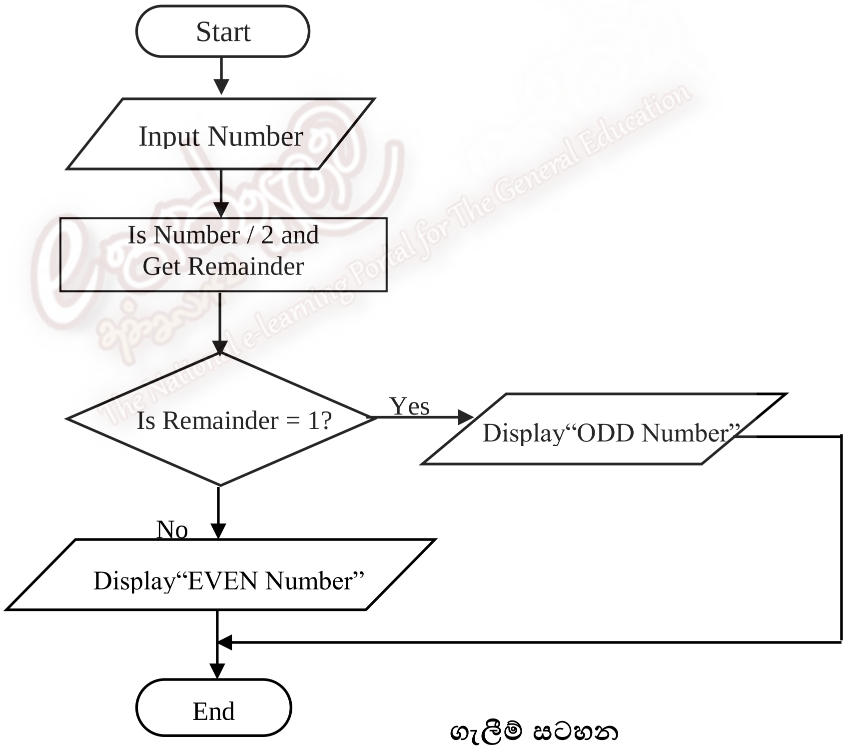
➢ සම්මත තේරීම් (වරණ) පාලන ව්‍යුහය

- අදාල කොන්දේසිය තෘප්ත වන්නේ නම් ප්‍රකාශය_1ද, කොන්දේසිය තෘප්ත නොවේ නම් ප්‍රකාශය_2ද ක්‍රියාත්මක කරනු ලබයි.




```
IF  < Condition >  THEN
        Statement_1
ELSE
        Statement_2
END IF
```

උදා:- 3. සංඛ්‍යාවක් ආදානය කල වට එමසංඛ්‍යාව ඔත්තේ සංඛ්‍යාවක්ද, ඉරට්ටේ සංඛ්‍යාවක්ද යන්න සෙවීම



ගැලීම් සටහන

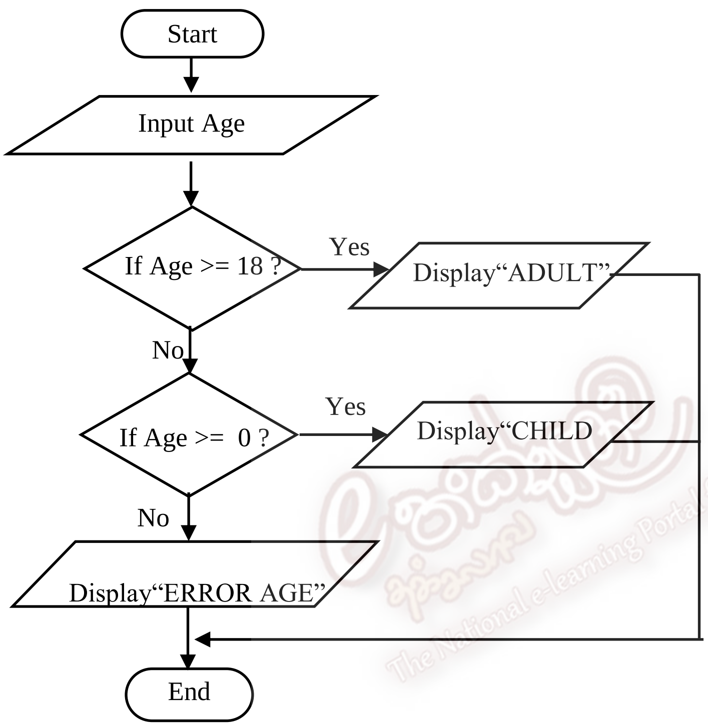
Begin
  Input Number,
    If Number Mod 2 = 1 then
      Display = "Number is Odd Number"
    Else
      Display = "Number is Even Number"
    End if
 End
	program two(input/output);
	var Number: Integer;
	begin
	    Writeln('Enter Your Number = ');
	    Readln(Number);
	    If Number Mod 2 = 1 then
	        writeln('Number is Odd Number')
	    else
	        writeln('Number is Even Number')
	    readln;
	end.

ව්‍යාජ කේතය	පැස්කල් ක්‍රමලේඛය

➢ කේවල විචල්‍යයකට බහු කොන්දේසි ඇති විට NESTED IF භාවිත වන ආකාරය

එක් විචල්‍යයක් කොන්දේසි කිහිපයක් මත තේරීමකට ලක් වන අවස්ථාවක දී IF ප්‍රකාශනය කොන්දේසි ගණනට අදාළ වාර ගණනක් යෙදීමට සිදුවේ. මෙහිදී IF …THEN… ELSE… ENDIF තේරීම සිදු කරන වාර ගණනට සමාන වාර ගණනක් යෙදිය යුතුය. මෙය Nested IF ප්‍රකාශ භාවිතය ලෙස හැඳින්වේ.

උදා:- 4. පුද්ගලයෙකුගේ වයස ආදානය කර ඇති විට දී ඔහු ගේ වයස් කාණ්ඩය තේරීම.



ගැලීම් සටහන

```
Begin
     Input Age,
         If Age  >= 18   then
                  Age Group = “Adult”
         Else
                 If Age  > 0   then
                       Age Group = “Child”
                 Else
                       Age Group = “Error”
                 End if
         End if
     Display  Age Group
End
```

ව්‍යාජ කේතය

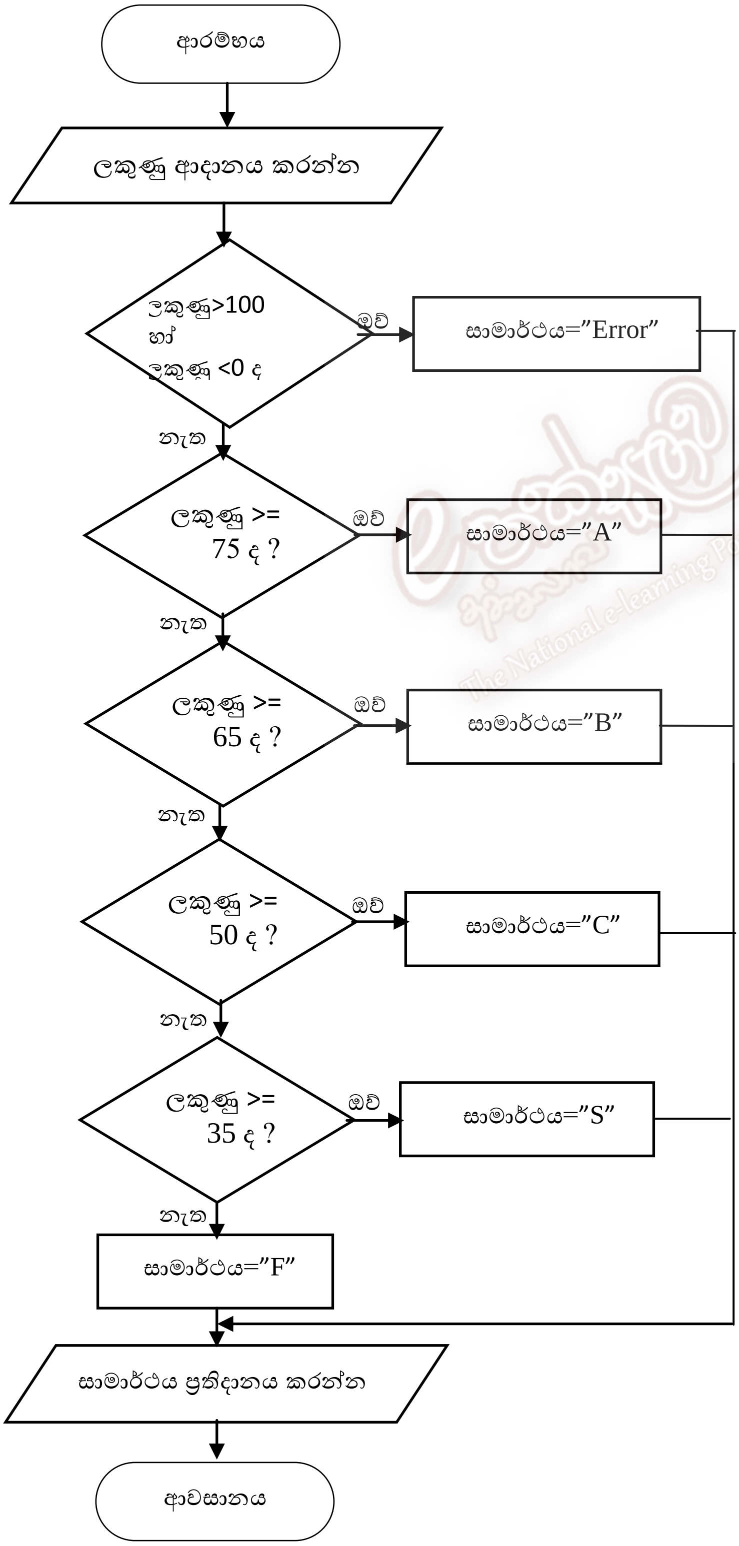
```
program three(input/output);

var   Age : Integer;
      Age_Group :String;
begin
         Writeln('Enter Your Age = ');
         Readln(Age);
         if Age >= 18 then
                  Age_Group= “ADULT”
         else
                  if Age >= 0 then
                           Age_Group= “CHILD”
                  else
                           Age_Group= “Error Age”
        Writeln('Your age group is ', Age_Group );
         readln;
end.
```

මේ අනුව තේරීම් පාලන ව්‍යුහය හෙවත් වරණය-Selection භාවිත වන ක්‍රමලේඛ ආකාර දෙකකින් දැක්විය හැක.

1. සරල තේරීම් පාලන ව්‍යුහය-Simple Selection
2. බහු තේරීම් පාලන ව්‍යුහය-Multiple Selection

උදා 5. කිසියම් විෂයකට සිසුවෙකු ලබාගත් ලකුණු ආදානය කල විට ඊට අදාල ශ්‍රේණිය පෙන්නුම් කිරීම.




```
Begin
input Marks
if Marks>100 or if Marks <0 then
        Grade =”Error”
else
        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”
                                end if
                        end if
                end if
        end if
end if
Display Grade
End.
```

ව්‍යාජ කේතකය


```
program Four_Grade(input/output);
var  Marks : integer;
     Grade : string;
begin
        writeln(‘Enter Marks = ’);
        read(Marks);
        if Marks>100 or  if Marks < 0 then
                Grade =”Error”
        else
        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(‘Your grade is = ’, Grade);
        readln;
End.
```

පැස්කල් ක්‍රමලේඛය

ගැලීම් සටහන

❖ කේවල විචල්‍යයකට බහු කොන්දේසි ඇති විට පහසුවෙන් පැස්කල් ක්‍රමලේඛය නිර්මාණය කිරීම සඳහා Case Statement භාවිත වන ආකාරය (ඉහත උදාහරණය යොදා ගෙන ඇත)
මෙහි දී IF ….THEN… ELSE… END IF **නැවත නැවත යොදා ගැනීමට අවශ්‍ය නොවේ.**

```
program Find_Grade(input,output);
var     Marks : integer;
        Grade: string;
Begin
        Weriteln('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
                Grade := 'Invalid Marks');
        End;
        Writeln('Your Grade is =' , Grade);
        readln;
End.
```

➢ පුණරාවර්තන/පුනර්කරන පාලන ව්‍යුහය සහිත ක්‍රමලේඛ- (Repetition/Iteration)

යම් උපදෙස් මාලාවක් යම් වාර ගණනක් නැවත නැවත භාවිත කිරීමට පුණරාවර්තන ප්‍රකාශන යොදා ගැනේ.

• පුනර්කරණ වාර ගණන නිශ්චිතව දන්නා අවස්ථාවලදී භාවිත කරන පුනර්කරණ ක්‍රමලේඛ

❖ <u>FOR – DOව්‍යුහය (පළමු ක්‍රමය)</u>

Syntax : For විචල්‍යය := ආරම්භක අගය to අවසාන අගය do
FOR <Variable Name> := <Value_1> TO < Value_2 > do

- මෙහි විචල්‍යය/Variable , Value_1 සහ Value_2 වල දත්ත ප්‍රරූපය පූර්ණ සංඛ්‍යා-(integer) ආකාරයේ විය යුතුය.
- පුනර්කරණය ආරම්භවීම සඳහා Value_1 හි අගයට වඩා Value_2 හි අගය වැඩි විය යුතුය.
- පුනර්කරණය Value_1 වලින් ආරම්භ වී වන අතර Value_2 වලින් අවසන් කෙරේ.
- එබැවින් FOR – DO ව්‍යුහයක් භවිත කල හැකි වන්නේ පුනර්කරණය වන වාර ගණන නිශ්චිතව දන්නා ක්‍රමලේඛ ගොඩනැගීමට පමණි.

පුනර්කරණ ව්‍යුහය	ආරම්භක අගය Value_1	අවසාන අගය Value_2	පුනර්කරණය වන වාර ගණන
FOR X:= 1 TO 5 DO	1	5	5
FOR X:= 0 TO 4 DO	1	4	5
FOR X:= 5 TO 10 DO	5	10	6

උදා **6. :** 1 සිට 10 දක්වා සංඛ්‍යා මුද්‍රණය කිරීම

```
Begin
        For  Number := 1 to 10 do
                Print Number
        Next Number
End.
```

ව්‍යාජ කේතය

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

පැස්කල් ක්‍රමලේඛය

- මෙහිදී Number විචල්‍ය අගය 1 සිට 10 දක්වා අනුයාත ලෙස වෙනස් වෙමින් ප්‍රතිදානය කරන අතර පුනර්කරන වාර දහයක් ක්‍රියාත්මක වේ.

❖ <u>FOR – DO ව්‍යුහය (දෙවන ක්‍රමය)</u>

Syntax : For විචල්‍යය := ආරම්භක අගය DOWNTO අවසාන අගය do
FOR <Variable Name> := <Value_1> DOWNTO < Value_2 > do

- පුනර්කරණය ආරම්භවීම සඳහා Value_1 හි අගය Value_2 හි අගයට වඩා වැඩි විය යුතුය.
- පුනර්කරණය Value_1 වලින් ආරම්භ වන අතර Value_2 වලින් අවසන්වේ.

පුනර්කරණ ව්‍යුහය	ආරම්භක අගය Value_1	අවසාන අගය Value_2	පුනර්කරණය වන වාර ගණන
FOR X:= 10 DOWNTO 5 DO	10	5	6
FOR X:= 4 DOWNTO 0 DO	4	0	5

උදා **7 :** 10 සිට 1 දක්වා අගයන් ප්‍රතිදානය කිරීම

```
Begin
        For  Number := 10 DOWNTO 1 do
                Print Number
        Next Number
End.
```

ව්‍යාජ කේතය

```
Program print_ReverseNos(input,output);
Var     Number: integer;
Begin
        For  Number := 10 DOWNTO 1 do
        Writeln(Number);
        readln;
End.
```

පැස්කල් ක්‍රමලේඛය

මෙහිදී Number විචල්‍යයේ අගය 10 සිට 1 දක්වා අනුයාත ලෙස වෙනස් වෙමින් ප්‍රතිදානය කරන අතර වාර දහයක් පුනර්කරනය ක්‍රියාත්මක වේ.

උදා 8: සංඛ්‍යා දහයක් ආදානය කර එකතුව සහ සාමාන්‍යය සෙවීම

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

❖ **For - Do ව්‍යුහයට අදාල පහත දී ඇති පැස්කල් ක්‍රමලේඛ අනුව තාරකා ත්‍රිකෝණ රටා සෑදෙන ආකාරය නිරීක්ෂණය කර එම ක්‍රමලේඛ ක්‍රියාත්මක වන ආකාරය තේරුම් ගැනීමට උත්සාහ කරන්න.**

```
program star1(output);
var x,y:byte;
const st=10;
begin
        for x:=1 to st do
        begin
            for y:=1 to x do
                write('*');
            writeln;
        end;
readln;
end.
```

පැස්කල් ක්‍රමලේඛ අංක -1

```
program star2 (output);
var x,y:byte;
const st=10;
begin
      for x:=st downto 1 do
      begin
          for y:=x downto 1 do
              write('*');
          writeln;
      end;
readln;
end.
```

පැස්කල් ක්‍රමලේඛ අංක - 2

```
program star3(output);
var x,y:byte;
const st=30;
begin
    for x:=st downto 1 do
    begin
        for y:=(2*x div 2) downto 1 do
            write(' ');
        for y:=1 to (st-x) do
        begin
            write(' ');
            write('*');
        end;
        writeln;
    end;
readln;
end.
```

```
program star4 (output);
var x,y:byte;
var Z:integer;
const st=10;
begin
    for x:=st downto 2 do
    begin
        for y:=1 to x-1 do
            write(' ');
        for y:=1 to 2*(st-y)-1 do
            write('*');
            writeln;
    end;
    readln;
end.
```

- **පුනර්කරණ වාර ගණන නිශ්චිතව නොදන්නා අවස්ථාවලදී භාවිත කරන පුනර්කරණ ක්‍රමලේඛ**

❖ While Do ව්‍යුහය.

- පුනර්කරණය ආරම්භයේදීම කොන්දේසිය පරීක්ෂා කර කොන්දේසිය සත්‍ය වන්නේනම් පමණක් පුනර්කරණය ආරම්භ වේ. කොන්දේසිය අසත්‍ය වන්නේ නම් කිසිවිටෙක පුනර්කරණය ආරම්භ නොවේ.
- පුනර්කරණය අවසන් වීම සඳහා පරීක්ෂා කරන ලද කොන්දේසිය අසත්‍ය විය යුතුම වේ.
- පුනර්කරණය සිදු වන විට කොන්දේසිය අසත්‍ය නොවේ නම් අපරිමිත වාර ගණනක් පුනර්කරණය වේ.(infinite loop)

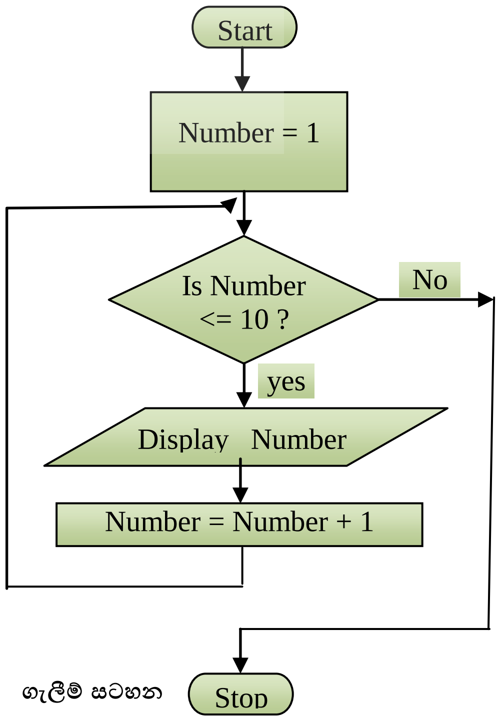
උදා 9 : 1 සිට 10 දක්වා සංඛ්‍යා ප්‍රතිදානය කිරීම.

```
Begin
      Number=0
      while Number <=10
                  Display Number
                  Number = Number     + 1
      End while
End.
```

ව්‍යාජ කේතය

```
program Display_10Nos(input/output)
var Number :integer;
Begin
      Number:=1
      while Number <=10
      begin;
            writeln(Number)
            Number: = Number  +  1
      End;
      readln
End.
```

පැස්කල් ක්‍රමලේඛය

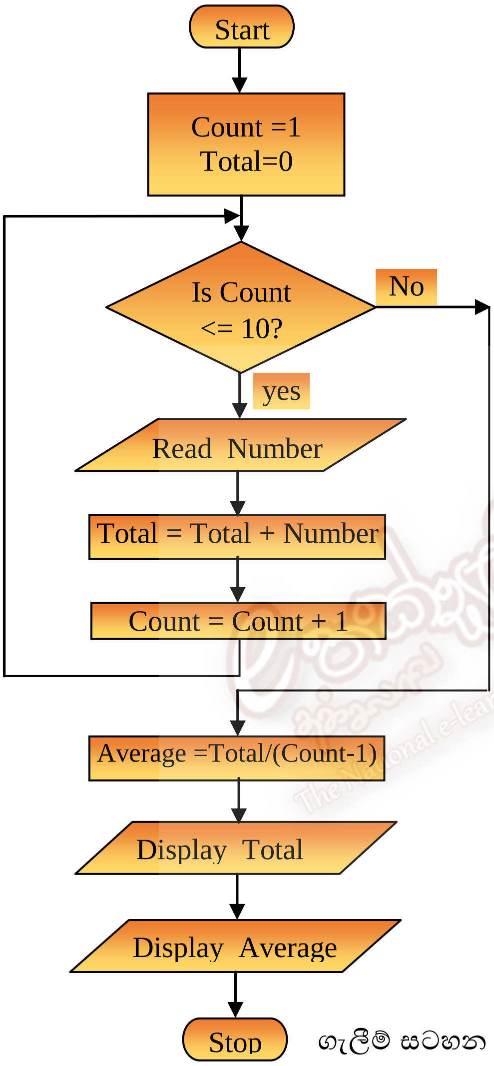


ගැලීම් සටහන

ඉහත උදාහරණයට අනුව

Number විචල්‍යයේ ආරම්භක අගය 1 බැවින් කොන්දේසිය සත්‍ය වේ. එවිට පුනර්කරණය ආරම්භ වන අතර පුනර්කරණය වන සෑම වාරයක් පාසාම Number හි අගයට 1 ක් එකතු වෙමින් Number විචල්‍යයෙහි අගය 10 ට සමාන වන තෙක් පුනර්කරණය සිදුවේ. ඒ අනුව Number විචල්‍යයෙහි අගය 10 හෝ 10 ට අඩුව ඇති විට පුනර්කරණය දැකිය හැක. Number විචල්‍යයේ අගය 11 වූ විට පුනර්කරණය වීම නවතී. එබැවින් මෙහි ප්‍රතිදානය ලෙස 1 සිට 10 දක්වා වූ සංඛ්‍යා පෙන්වයි.

උදා 10: සංඛ්‍යා දහයක් ආදානය කර එකතුව සහ සාමාන්‍යය සෙවීම



ගැලීම් සටහන

```
Begin
    Count = 1
    Total = 0
    while Count<=10
            input (Number)
            Total=Total + Number
            Count = Count + 1
    End while
    Average=Total/(Count-1)
    Display Total
    Display Average
End.
```

ව්‍යාජ කේතය

```
program tot_avg(input/output)
var   Count: integer;
      Total: integer;
      Number: integer;
      Average: real
Begin
    Count := 1
    Total := 0
    while Count <= 10
    begin;
            writeln (‘Enter Number = ’)
            read(Number)
            Total=Total + Number
            Count = Count + 1
    End;
    Average=Total/(Count-1);
    writeln(‘Total is = ’,Total);
    writeln(‘Average is = ’,Average);
End.
```

පැස්කල් ක්‍රමලේඛය

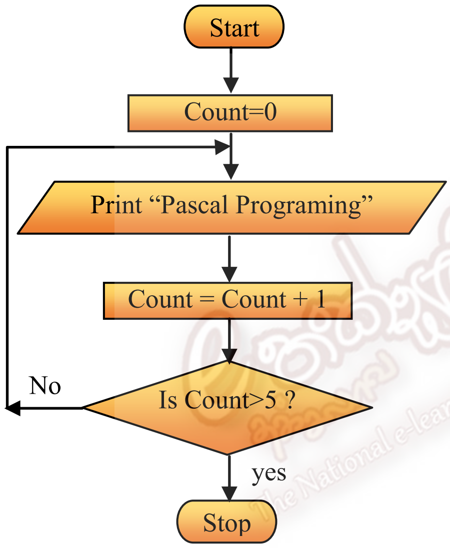
➢ While - Do ව්‍යුහයට අදාල මෙම පැස්කල් ක්‍රමලේඛය ක්‍රියාත්මක කර ප්‍රතිදානය නිරීක්ෂණය කර එම ප්‍රතිදානය ලැබෙන ආකාරය තේරුම් ගැනීමට උත්සාහ කරන්න.

```
program test_star();
Uses Crt;
var line,stars:integer;
begin
textbackground (green);
textcolor(blue);
        line:=1;
        while line<=20 do
        begin
             stars:=1;
             while stars<=line do
             begin
                    write('*');
                    stars:=stars+1;
             end;
        line:=line+1;
        writeln;
        end ;
```

❖ Repeat Until ව්‍යුහය

- පුනර්කරණය වීමට ඇති ප්‍රකාශ එක් වරක් ක්‍රියාත්මක වීමෙන් පසුව කොන්දේසිය පරීක්ෂා කර කොන්දේසිය අසත්‍ය වන්නේ නම් පමණක් පුනර්කරණය ආරම්භ වී කොන්දේසිය සත්‍ය වීමත් සමගම පුනර්කරණය අවසන් කරයි.
- පුනර්කරණය වන අතර වාරයේදී කොන්දේසිය සත්‍ය නොවේ නම් අපරිමිත වාර ගණනක් පුනර්කරණය වේ.(infinite loop)

උදා. 11: “Pascal Programing” යන වචනය කිහිප වරක් මුද්‍රණය කර ගැනීම.




```
Begin
   Count=0;
   Repeat
         Print “Pascal Programing”
         Count:= Count +1
   Until Count>5
end.
```

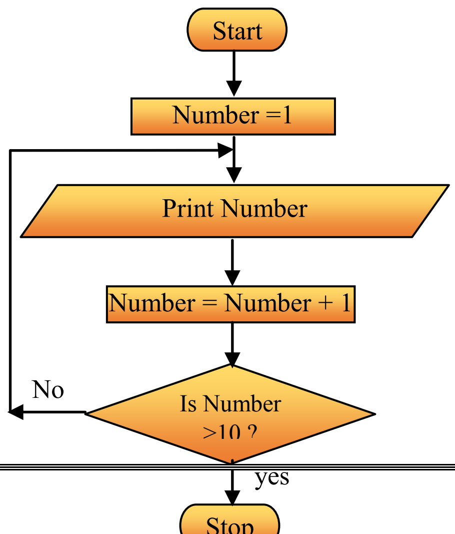
ව්‍යාජ කේතය

```
program Print_Pascal(input/output)
var   Count: integer;
Begin
   Count=0;
   Repeat
         writeln(‘Pascal Programing’);
         Count:= Count +1
   Until Count>5;
end.
```

පැස්කල් ක්‍රමලේඛය

- ➢ ක්‍රමලේඛය ක්‍රියාත්මක වන විට මුලු අවස්ථාවේ Count විචල්‍යයේ ආරම්භක අගය 0 වේ. එවිට ‘Pascal Programing’ යන වචනය තිරය මත දර්ශනය වේ. පසුව Count විචල්‍යයේ අගයට 1 ක් එකතු වී Count විචල්‍යයේ අගය 5 ට විශාල වේ දැයි පරීක්ෂා කරයි. මෙසේ Count විචල්‍යයේ අගය 5 වන තෙක් පුනර්කරණය සිදුවන අතර Count හි අගය 6 වූ විට පුනර්කරණය වීම නවතී.
- ➢ ඒ අනුව පුනර්කරණය නවතින විට ‘Pascal Programing’යන වචනය තිරය මත 6 වරක් දර්ශනය වේ.

උදා. 12: 1 සිට 10 දක්වා වූ සංඛ්‍යා මුද්‍රණය කර ගැනීම.




```
Begin
   Number=1
   Repeat
         Print Number
         Number= Number +1
   Until Number>10
end.
```

ව්‍යාජ කේතය

```
program Print_Number(input/output)
var   Number: integer;
Begin
   Number=1;
   Repeat
         writeln(Number);
         Number:= Number +1
   Until Count>10;
```

ඉහත උදාහරණයට අනුව

Number විචල්‍යයේ ආරම්භක අගය 1 වේ. පුනර්කරණය වීමට ඇති ප්‍රකාශ එක් වරක් ක්‍රියාත්මක කරයි. එවිට print Number ප්‍රකාශයට අනුව Number විචල්‍යයේ ආරම්භක අගය වන 1 මුද්‍රණය වේ. පසුව කොන්දේසිය පරීක්ෂා කර අසත්‍ය නම් පුනර්කරණය ආරම්භ වන අතර පුනර්කරණය වන සෑම වාරයක් පාසාම Number හි අගයට 1 ක් එකතු වෙමින් Number විචල්‍යයෙහි අගය 10 වන තෙක් පුනර්කරණය සිදුවේ. ඒ අනුව Number විචල්‍යයෙහි අගය 10 හෝ 10 ට අඩුව ඇති විට පුනර්කරණය දැකිය හැක. Number විචල්‍යයේ අගය 11 වූ විට පුනර්කරණය වීම නවතී. එබැවින් මෙහි ප්‍රතිදානය ලෙස 1 සිට 10 දක්වා වූ සංඛ්‍යා පෙන්වයි.

10. නීඩිත පාලන ව්‍යුහ

තේරීම හෙවත් වරණය හෝ පුනර්කරණය යන එකිනෙක වෙනස් පාලන ව්‍යුහ ඇල්ගොරිතමයේ (ගැටළුවේ) ස්වභාවය මත එකට සම්බන්ධ කර භාවිත කිරීමට සිදුවන අවස්ථා පවතී. මෙහිදී වරණයක් තුල පුනර්කරණයක් සිදුවීම හෝ පුනර්කරණය සිදුවන අතර තුර වරණයකට අදාල කොන්දේසි පදනම් කර ගත් ක්‍රියාවලින් සිදුවිය හැකිය. මෙවැනි තත්වයන් යටතේ ගොඩනගනු ලබන ක්‍රමලේඛ නීඩිත පාලන ව්‍යුහය සහිත ක්‍රමලේඛ ලෙස හැඳින්වේ.

❖ වරණය තුල පුනර්කරණය භාවිතය.

- වරණයක කොන්දේසියක් තෘප්ත වීම හෝ නොවීම අනුව පුනර්කරණයක් සිදුවිය හැක.

උදා. 13:

ශ්‍රී ලංකා තොරතුරු තාක්ෂණ විද්‍ය පීඨයට ඇතුළත් වීම සදහා අයදුම් කර ඇති සිසුන් අතරින් එහි උසස් තොරතුරු තාක්ෂණ ඩිප්ලෝමා පාඨමාලාව හැදෑරීමට තෝරා ගනු ලබන්නේ

- අ.පො.ස (උ.පෙළ)විභාගය සමත්ව තිබීම සහ
- අනතුරුව විද්‍යාපීඨය මගින් පව.ත්වන බහුවර්ණ ප්‍රශ්න 25 සමන්විත මාර්ග ගත පරීක්ෂණයක සියලුම ප්‍රශ්න වලට නිවැරදි පිළිතුරු සපයා තීබිම යන සාධක දෙකම සපුරා ඇති විටය.

අ.පො.ස(උ.පෙළ) විභගය සමවත් ව සිටියත් මාර්ග ගත පරීක්ෂණයේ (On-Line Test) සියලුම ප්‍රශ්න වලට නිවැරදිව පිළිතුරු දීමට අපොහොසත් වූ සිසුන් උසස් තොරතුරු තාක්ෂණ ඩිප්ලෝමාවට තෝරා නොගැනේ. විද්‍යාපීඨයේ උසස් තොරතුරු තාක්ෂණ පාඨමාලාවට සිසුන් තෝරා ගැනීම පිළිබඳ ඇල්ගොරිතමය පහත ගැලීම් සටහනේ දැක්වේ.

ගැලීම් සටහනේ සමහර ඉංග්‍රීසි යෙදුම් වලට අදාලව සිංහල අර්ථ දැක්වීම පහත පරිදි වේ.

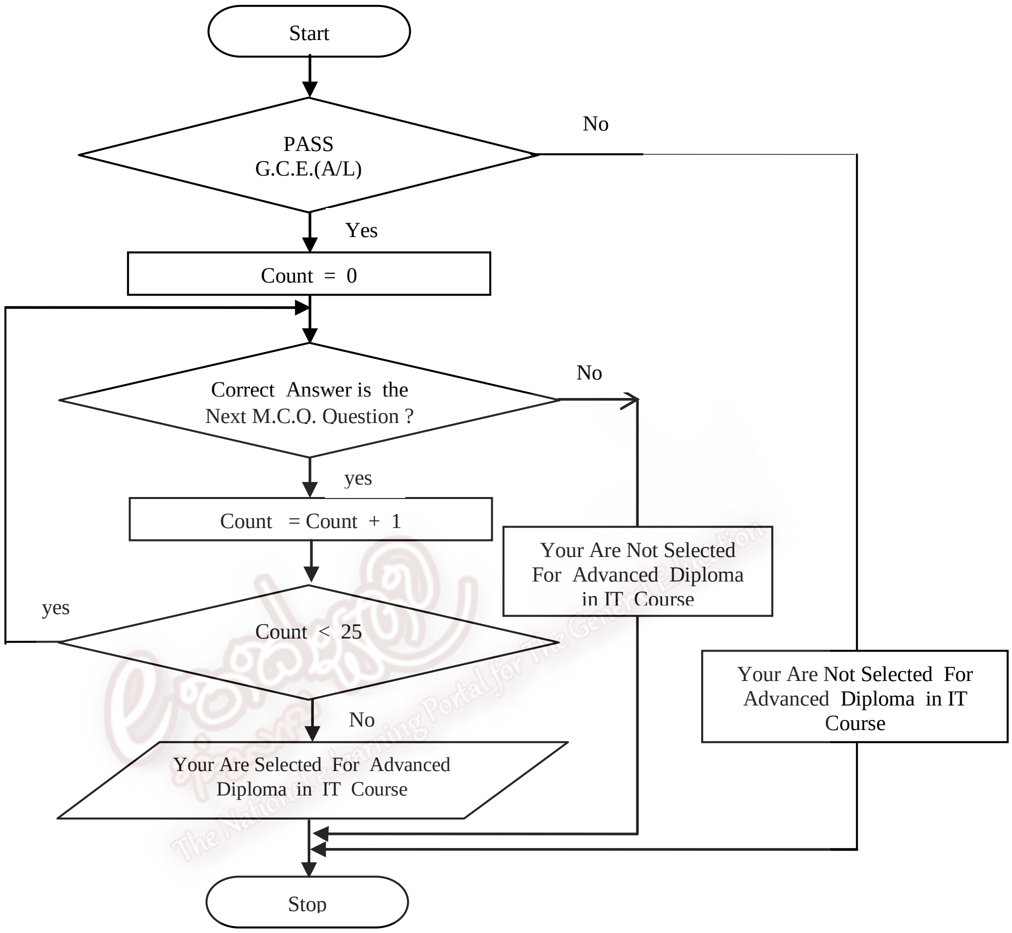
- අ.පො.ස (උ.පෙළ)විභාගය සමත් ද
  PASS G.C.E.(A/L) Examination ?

- බහුවර්ණ ප්‍රශ්න වලට පිළිතුරු දුන් වාර ගණන
  Count = 0

- ඊළඟ ප්‍රශ්නයට දුන් පිළිතුර නිවැරදි ද
  Correct Answer is the Next M.C.Q. Question ?

- ඔබ උසස් තොරතුරු තාක්‍ෂණ ඩිප්ලෝමා පාඨමාලාවට තේරී ඇත.
  Your Are Selected For Advanced Diploma in I T Course

- ඔබ උසස් තොරතුරු තාක්‍ෂණ ඩිප්ලෝමා පාඨමාලාවට තේරී නැත.
  Your Are Not Selected For Advanced Diploma in I T Course



- මෙම ගැලීම් සටහනට අදාල ව්‍යාජ කේතය සහ පැස්කල් ක්‍රමලේඛය ඉදිරිපත් කරන්න.

ව්‍යාජ කේතය:	පැස්කල් ක්‍රමලේඛය
....	....
....	....
....	....
....	....
....	....
....	....
....	....
....	....
....	....
....	....
....	....
....	....
....	....
....	....
....	....
....	....
....	....
....	....
....	....
....	....

උදා. 14: පරිශීලකගේ තේරීම අනුව ප්‍රතිදානය ආරෝහණ හෝ අවරෝහණ සංඛ්‍යා අනුක්‍රමයක් ලබා ගැනීම.

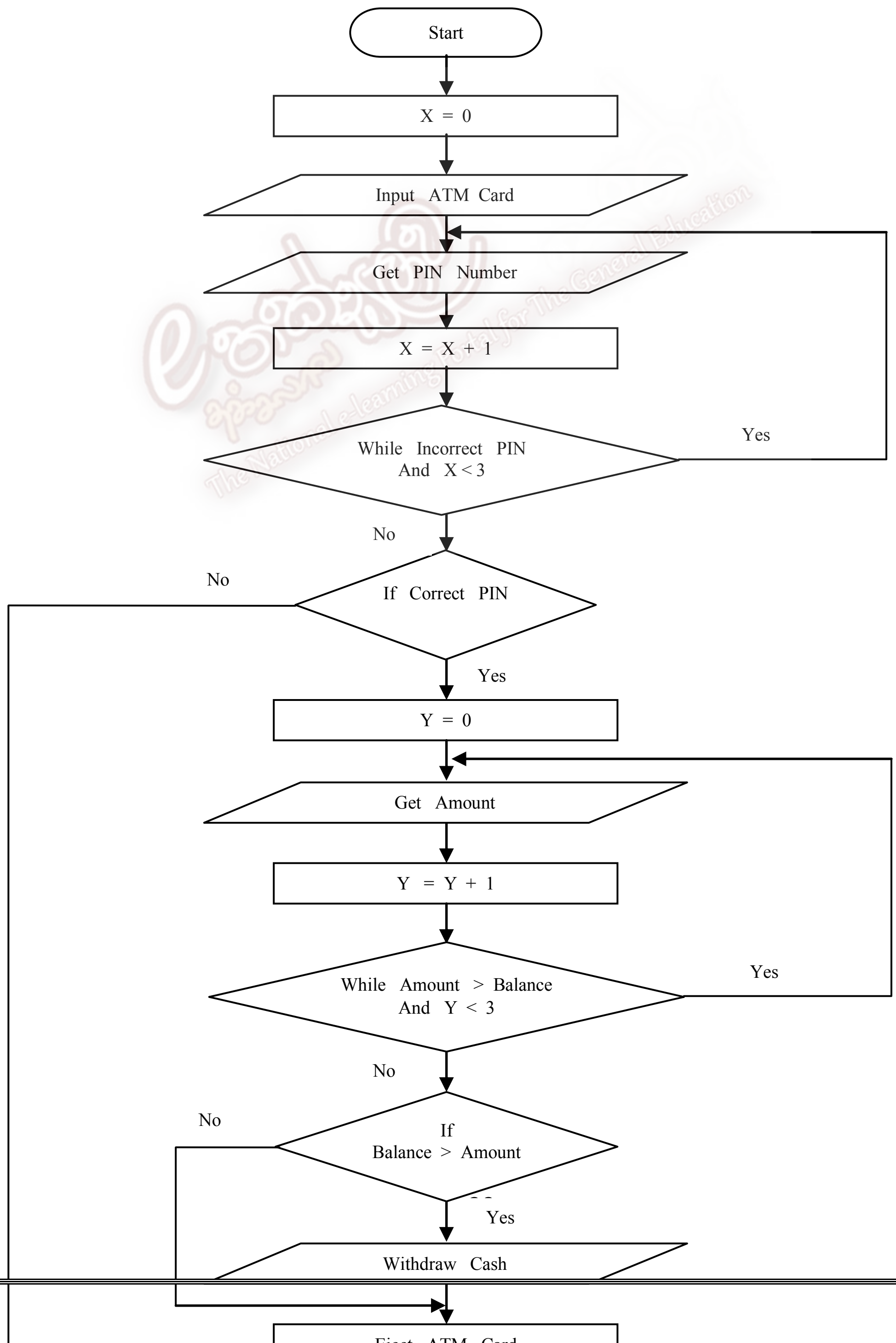
මෙහි පැස්කල් ක්‍රමලේඛය පහත දැක්වේ. ඊට අදාල ගැලීම් සටහන ඇඳ සහ ව්‍යාජ කේතය ලියා දක්වන්න.

```
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;
       readln;
end.
```

ගැලීම් සටහන:	ව්‍යාජ කේතය:
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....

උදා. 15: ස්වයංක්‍රීය ටෙලර් යන්ත්‍ර කාඩ් පතක් (ATM Card) භාවිත කරමින් තම බැංකු ගිණුමෙන් **මුදල් ලබා ගැනීමේ කාර්යය පමණක්** සිදු කරන අවස්ථාවක් නිරූපනය කරන ඇල්ගොරිතමයකට අදාල ගැලීම් සටහනක් පහත දැක්වේ. මෙම සිද්ධියට සඳහා දී ඇති ගැලීම් සටහන ඉතා හොඳින් අධ්‍යයනය කර එහි තර්කයට අදාල

i. ව්‍යාජ කේතය
ii. පැස්කල් ක්‍රමලේඛය ලියා දක්වන්න.



ව්‍යාජ කේතය:	පැස්කල් ක්‍රමලේඛය :

❖ **පුනර්කරණය තුල වරණය භාවිතය.**

උදා. 16: කිසියම් සංඛ්‍යා 10ක් කියවා එක් එක් සංඛ්‍යාව ධන සංඛ්‍යාවක් (positive number) ද ඍණ සංඛ්‍යාවක් (negative number) ද එසේ නොමැති නම් බිංදුව (zero) ද යන්න මුද්‍රණය කිරීම සඳහා ලියන ලද ව්‍යාජ කේතය (pseudo code) පහතින් දැක්වේ.

1. මෙම ව්‍යාජ කේතය සඳහා ගැලීම් සටහන (flow chart) ඇඳ දක්වන්න.
2. මෙම සංසිද්ධිය සඳහා වූ පැස්කල් ක්‍රමලේඛය නිර්මාණය කරන්න.

```
Begin
        Count = 1
        While Count  <=  10
               Read  Number
               If  Number  = 0 Then
                       Result = “Number is zero”
               Else
                       If  Number > 0 Then
                              Result = “Number is positive Number.”
                       Else
                              Result= “Number is negative Number.”
               Endif
               Endif
               Print Number, Result
               Count = Count + 1
        End While
End.
```


ගැලීම් සටහන:	පැස්කල් ක්‍රමලේඛය
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....

උදා. 17: එක්තරා බැංකුවකින් දෙසැම්බර් මාසය සඳහා ණය ඉල්ලුම් කර ඇති අයදුම්කරුවන් සමූහයකට ඔවුන්ගේ මාසික ආදායම අනුව ලබා දිය හැකි ණය මුදල සංදර්ශනය කිරීමට අදාල ඇල්ගොරිතමයක ව්‍යාජ කේතය-(Pseudo Code) පහත දැක්වේ.

```
Start
    No. of  Applicant  = 1
    Do While No. of Applicant  <=  50
        Input  Name, Monthly Income
        If  Monthly Income >= 25000 then
            Loan Amount  = 100000
        else
            If  Monthly Income >= 20000 then
                Loan Amount  = 75000
            else
                If  Monthly Income >= 15000 then
                    Loan Amount  = 50000
                else
                    Loan Amount  = 0
                end if
            end if
        end if
        Display “Loan Amount”
        No. of Applicant = No. of Applicant + 1
    End While
End.
```

ඉහත ව්‍යාජ කේතයට අදාල ගැලීම් සටහන හා පැස්කල් ක්‍රමලේඛය නිර්මාණය කරන්න.

ගැලීම් සටහන:	පැස්කල් ක්‍රමලේඛය

උදා. 18: පරිශීලක විසින් ආදානය කරනු ලබන සංඛ්‍යා ඔත්තේ සංඛ්‍යාද? , ඉරට්ටේ සංඛ්‍යාද? යන්න හඳුනා ගැනීම හා ඔත්තේ හා ඉරට්ටේ සංඛ්‍යා ප්‍රමාණ වෙන වෙනම ගණනය කිරීම.

- මෙම ඇල්ගොරිතමයට අදාල පැස්කල් ක්‍රමලේඛය පහත දැක්වේ. ගැලීම් සටහන ඇඳ ව්‍යාජකේතය ලියා දක්වන්න.

```
program rep_sel(input,output);
var num,rem,count,e_count,o_count:integer;
begin
    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;
    write('Even Numbers are ');
    writeln(e_count);
    write('Odd Numbers are ');
    writeln(o_count);
    readln;
end.
```

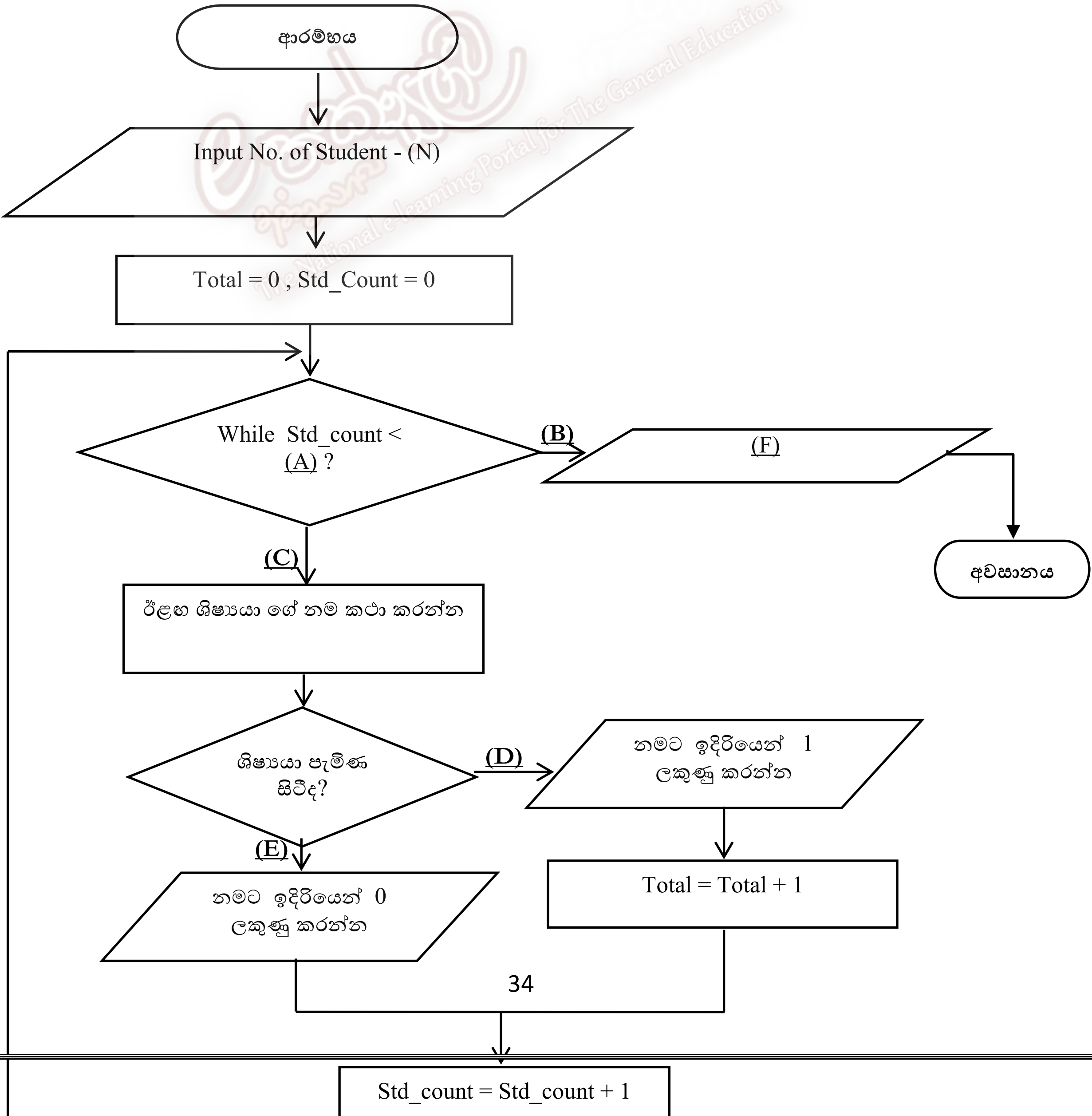
ගැලීම් සටහන:	ව්‍යාජ කේතය ....

උදා. 19: **විද්‍යොදය** විද්‍යාලයේ 11 ශ්‍රේණියේ නාම ලේඛණයේ නම් සඳහන් සිසුන් 40 දෙනෙකු ගේ පැමිණීම සටහන් කරන ආකාරය දැක්වෙන ඇල්ගොරිතමයක් පහත ගැලීම් සටහනෙන් පෙන්නුම් කරයි. පන්ති භාර ගුරු මහතා පන්තියට ගොස් නාම ලේඛනයත් පන්තියෙහි සිසුන් සංඛ්‍යාව සොයා එය ආදානය කරයි. සෑම සිසුවෙකුගේම නම කියවා සිසුවා පැමිණ සිටියි නම්, නාම ලේඛනයේ ඔහුගේ නමට ඉදිරියෙන් 1 ලකුණු කරන අතර, පැමිණ නැතිනම් 0 ලකුණු කරයි. මෙම ක්‍රියාදාමය පන්තියේ සියළුම සිසුන් සඳහා සිදු කර අවසානයේදී පැමිණ සිටි සිසුන් සංඛ්‍යාවේ එකතුව සංදර්ශනය කරයි.

N – No. of Student - පන්තියේ සිටින ශිෂ්‍ය සංඛ්‍යාව
Total - පැමිණ සිටින සිසුන් සංඛ්‍යාවේ එකතුව

i. ගැලීම් සටහනේ A සිට F දක්වා හිස් තැන් වලට සුදුසු වගන්ති / වචන/ හෝ අගයන් ලියා දක්වන්න.

ii. ගැලීම් සටහනේ පෙන්නුම් කරන ඇල්ගොරිතමය සඳහා ව්‍යාජ කේතය සහ පැස්කල් ක්‍රමලේඛය ලියා දක්වන්න.



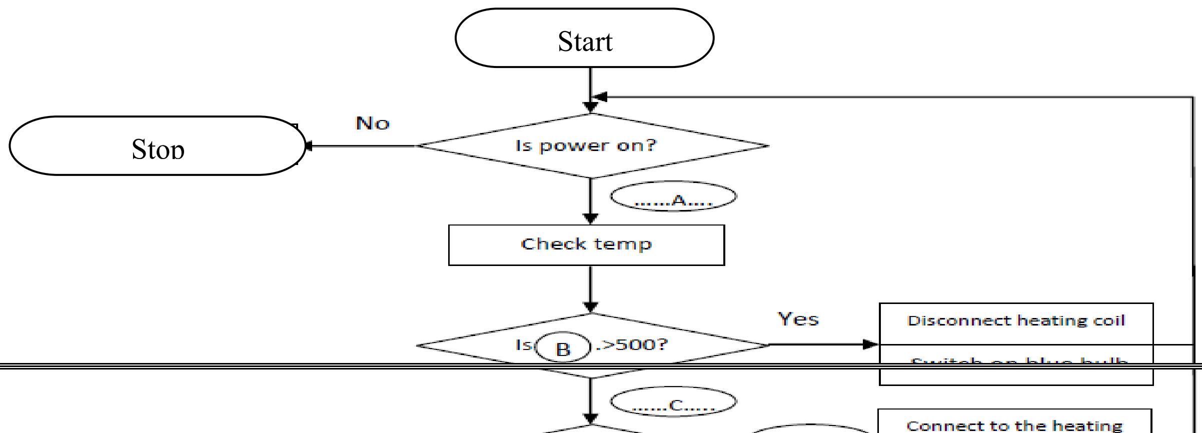
ව්‍යාජ කේතය:

පැස්කල් ක්‍රමලේඛය

උදා. 20: විදුලි ස්ත්‍රික්කයක උෂ්ණත්ව පාලන ක්‍රියාවට අදාලව ශිෂ්‍යයෙකු විසින් නිර්මාණය කල පරිපථයක් සඳහා වූ ගැලීම් සටහනක් පහත දැක්වේ.මෙහි කපු රෙදි මැදීම සඳහා උෂ්ණත්ව පරාසය 448^0 C සිට 500^0 C දක්වා වේ.

i. ගැලීම් සටහනේ A සිට E දක්වා හිස්තැන් පුරවන්න

ii. ගැලීම් සටහනට අදාල ව්‍යාජ කේතය සහ පැස්කල් ක්‍රමලේඛය ලියන්න.



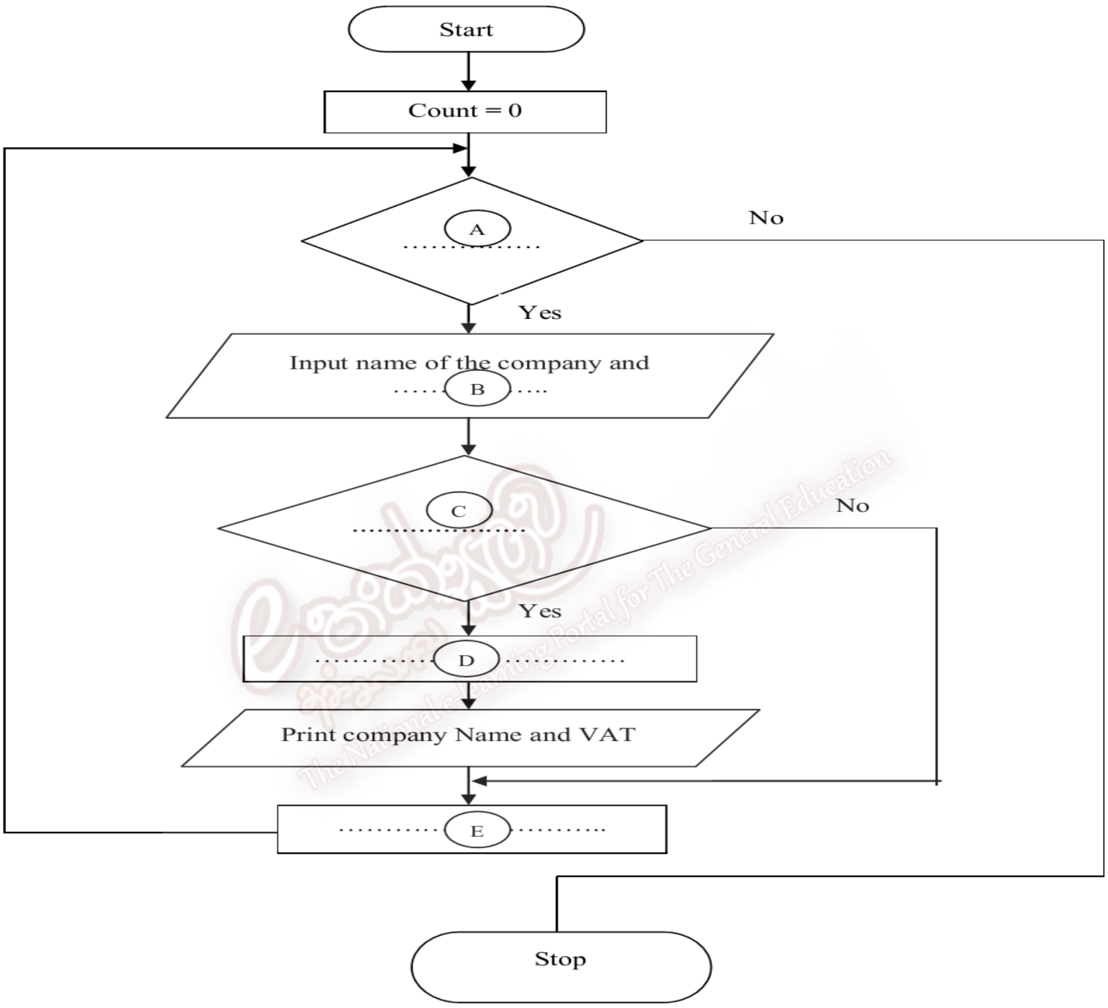
ව්‍යාජ කේතය:

පැස්කල් ක්‍රමලේඛය

උදා. 21: එක්තරා ප්‍රාදේශීය ලේකම් කාර්යාලයක ලියාපදිංචි වී ඇති ව්‍යාපාරික සමාගම් 50ක් අතරින් මසක ආදායම මිලියන 10 ට වැඩි සමාගම් වලින් එකතු කළ අගය මත බද්ද - (VAT) ලෙස මාසික ආදායමෙන් 12%ක මුදලක් අයකරගනු ලැබේ. බදු ගෙවිය යුතු සමාගම් හඳුනා ගැනීම සඳහා නිර්මාණය කිරීමට බලාපොරොත්තු වන පරිගණක ක්‍රමලේකයකට අදාළ ගැලීම් සටහනක් පහත දැක්වේ.

i. ඉහත ගැටළුව නිරූපණය සඳහා අඳින ලද අසම්පූර්ණ ගැලීම් සටහනක් පහත දැක්වේ. **A, B, C, D** හා **E** යන ලේබල සඳහා ගැලපෙන ප්‍රකාශ ලියා දක්වන්න. (ගැලීම් සටහන ඔබේ පිළිතුරු පත්‍රයේ පිටපත් කිරීම අවශ්‍ය නොවේ.)

ii. ගැලීම් සටහනට අදාල ව්‍යාජ කේතය සහ පැස්කල් ක්‍රමලේඛය ලියන්න.



ව්‍යාජ කේතය:	පැස්කල් ක්‍රමලේඛය
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....
	....

11. අරාව - (Arrays)

අරාවක් යනු විචල්‍ය කිහිපයක එකතුවක්. එකම ප්‍රරූපයට අයත් දත්ත එකම නමක් යොදාගනිමින් මතකය තුළ තැන්පත් කිරීමට අරාව භාවිත කෙරේ.

එකම ප්‍රරූපයට අයත් දත්ත එකම නමක් යොදාගනිමින් මතකය තුල තැම්පත් කිරීමට අරාව භාවිත කරයි. එබැවින් එක් එක් දත්තය සඳහා වෙන වෙනම විචල්‍ය නාමයන් වෙනුවට තනි විචල්‍ය නාමයක් යටතේ අවශ්‍ය ප්‍රමාණයකට දත්ත තැම්පත් කිරීමට අරාවක් භාවිත කෙරේ.

ක්‍රමලේඛනයේදී දත්ත මතකය තුල තැම්පත් කිරීමට විචල්‍ය භාවිත කිරීම අනිවාර්ය වේ. එමෙන්ම එම විචල්‍යක් සඳහා සුදුසු දත්ත ප්‍රරූපයක් පැවතිය යුතුය. එකම දත්ත ප්‍රරූපයට අයත් දත්ත තැම්පත් කිරීමට නමින් එකිනෙක වෙනස් විචල්‍ය සමූහයක් අවශ්‍ය වේ.

උදා:- පූර්ණ සංඛ්‍යා 5 ක් මතකය තුල තැම්පත් කිරීම සඳහා විචල්‍ය 5 ක් අවශ්‍ය වේ. එවැනි විචල්‍ය භාවිතයට පෙර පහත දැක්වෙන ආකාරයෙන් හඳුන්වාදිය යුතුය.

```
Var     a,b,c,d,e : integer;  හෝ
        N1,N2,N3,N4,N5 : real;
```

❖ ඒක-මාන අරාව

ඒක-මාන අරාව අනුක්‍රමික ලෙස නිශ්චිත ප්‍රමාණයක එකම ප්‍රරූපයකට අයත් දත්ත තැම්පත් කිරීම සඳහා භාවිත කරන දත්ත ව්‍යුහයකි.

අරාවක් මඟින් යාබද මතක අවකාශ සමූහයක් වෙන් කර ගනී.

ඒක-මාන අරාවක් පහත දැක්වෙන අයුරින් හඳුන්වා දිය හැක.

Syntax(කාරක රීතිය) :
Var <Name_of_Array> : array [first value .. last value] of data type

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

මෙහි දී පූර්ණ සංඛ්‍යාත්මක අගයන් 10 ක් ඇතුළත් කලහැකි marks නම් වූ අරාව සෑදේ.

❖ අරාවක ලක්ෂණ

- අරාවක කොටස් - (අවයව) යාබදව පිහිටයි.
- අරාවක දර්ශකය හෙවත් අනුපිළිවෙල අංකය අරාවේ නම සමඟ කොටු වරහන් තුල දක්වයි.

උදා(i):- Var X : Array[0..4] of integer; වන අරාවේ ප්‍රමාණය 5 කි.
එය X[0] සිට X[4] දක්වා අවයව 5 කින් සමන්විත වේ.

- කොටු වරහන තුල දර්ශකය දක්වනු ලබයි.
- අරාව හඳුන්වාදෙන ක්‍රමය අනුව එහි දර්ශක පිහිටන ස්ථානය වෙනස් වේ.

උදා(ii):- Var Y : Array[1..5] of integer;
එය Y[1] සිට Y[5] දක්වා අවයව 5 කින් සමන්විත වේ.

- එකම ප්‍රරූපයකට අයත් දත්ත පමණක් අරාවේ තැම්පත් කල හැකිය.
- අරාවේ ඕනෑම අවයවයක් වෙත අහඹු ලෙස පිවිසීමට හැකිය. එබැවින් පුනර්කරණ පාලන ව්‍යුහයක් මඟින් අරාවක් පහසුවෙන් පාලනය කල හැකිය.

උදා: i

X [0]	X [1]	X[2]	X [3]	X[4]

ආරම්භක අවයවය (අඩුම දර්ශකය)

අවසාන අවයවය (වැඩිම දර්ශකය)

උදා: ii

Y[1]	Y[2]	Y[3]	Y[4]	Y[5]

උදා. 22: පන්තියක සිටින සිසුන් 40 දෙනෙකුගේ ICT ලකුණු අරාවකට - Array ඇතුළත් කිරීම.

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

❖ අරාවකට අගයන් පැවරීම

	num[0]	num[1]	Num[2]	Num[3]	Num[4]
var num : array[0..4] of integer; →					
num[0]:= 50; →	50				
num[2]:=45,num[4]:=70; →	50		45		70
num[1]:=num[4] + 35; →	50	105	45		70
num[3]:=num[0]+num[2] →	50	105	45	95	70

❖ අරාවක අගයන් ප්‍රකාශයට පත්කිරීම.

```
writeln(num[3]);              - 4 වන අවයවය ප්‍රකාශයට පත්කිරීම (95)

writeln(num[2],num[4]);       - 3 , 5 වන අවයව ප්‍රකාශයට පත්කිරීම (45 , 70)

for  x := 0 to 3 do           - අරාවේ පළමු අවයව 4 ප්‍රකාශයට පත්කිරීම(50,105,45,95)
        writeln(num[x]);

for  x := 2 to 4 do           - අරාවේ 3,4,5 යන අවයව 3 ප්‍රකාශයට පත්කිරීම( 45,95,70)
        writeln(num[x]);

for  x := 0 to 4 do           - අරාවේ සියළුම අවයව  ප්‍රකාශයට පත්කිරීම( 50,105,45,95,70)
        writeln(num[x]);
```

උදා.23: පංතියක සිසුන් 40 දෙනෙකු ICT විෂයට ලබාගත් ලකුණු ඇතුළත් කර වැඩිම ලකුණ සහ ලකුණුවල සාමාන්‍ය අගය ගණනය කිරීම සඳහා වූ පැස්කල් ක්‍රමලේඛය.

```
program ictMarks(input,output);
var   marks : array[0..39] of  integer;
      i, total, max : integer;
      avg : real;
     begin
            for i:= 0 to 39 do
            begin
                  writeln('Enter ICT Marks = ');
                  read (marks[i]);   (* Read Marks to array *)
                   total := total + marks[i];  (* Add marks *)
           end;
                 avg := total/40;
                 max := marks[0];
                 for i :=1 to 39 do
                 begin
                         if marks[i] > max  then
                         max := marks[i];
                 end;
                 writeln('Maximum marks=', max);
                 writeln('Average marks=',avg);
           readln;
     end.
```

➢ **අරාව පදනම් කර ගත් සරල පැස්කල් ක්‍රමලේඛ කිහිපයක් පහත දී ඇත. එම ක්‍රමලේඛ වල සිදුවන ක්‍රියාවලිය පැහැදිලි කර ලැබෙන ප්‍රතිදානයන් ලියා දක්වන්න.**

(1).

```
var  students: array[1..5]  of  string[5];   {මෙහි string[5] නිසා ප්‍රතිදානය වචන 5 කට සීමා වේ}
var  num: byte;
begin
  for num:=1 to 5 do
          begin
                writeln('Input Student Name  = ');
                readln(students[num]);
```

```
        end;
        for num:=1 to 5 do
        writeln(students[x]);
    readln;
    end.
```

{ප්‍රතිදානය සිදුවන කොටස}

ක්‍රියාවලිය :-

....

....

....

....

....

....

....

ප්‍රතිදානය:-

....

....

(2).

```
    var  marks : array[1..5]  of   byte;
    var  x : byte;
    begin
        for  x := 1 to 5 do
        begin
            writeln('Input Value = ');
            readln(marks[x]);
        end;
        for x := 1 to 5 do
            writeln(marks[x]);
        readln;
    end.
```

{ප්‍රතිදානය සිදුවන කොටස}

ක්‍රියාවලිය :-

....

....

....

....

....

....

ප්‍රතිදානය:-

....

....

(3).

```
    var     Marks : array [1 .. 5] of Integer;
            num : Integer;
    Begin
        for num:= 1 to 5 do
        begin
            Write ('Input student number ', num, ' mark: ');
```

```
            readln(Marks[num]);
        End;
        for num:= 1 to 5 do
        begin
            Write('Student number ', num, ' mark is : ', Marks[i]);
            if Marks[num] >= 50 then
                Writeln(' Pass')
            else
                Writeln(' Fail');
        end;
        Writeln('Press enter key to close');
        Readln;
    end.
```

ක්‍රියාවලිය :-

....

....

....

....

....

....

....

....

....

....

ප්‍රතිදානය:-

....

....

....

....

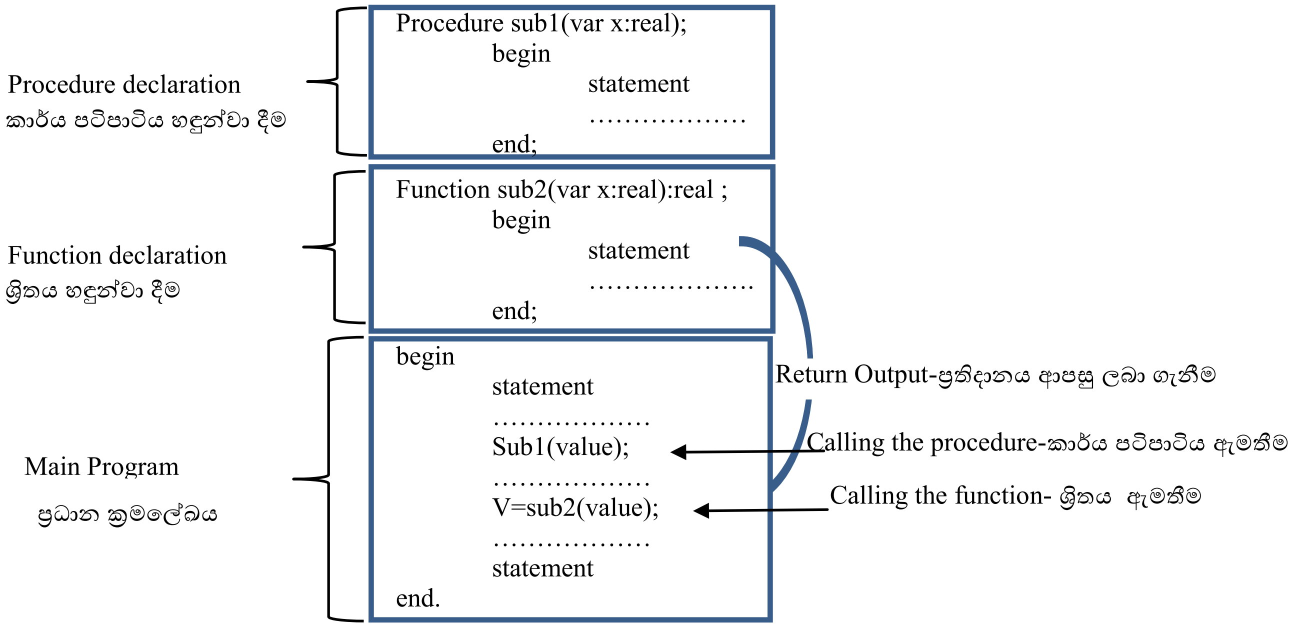
උප ක්‍රමලේඛ භාවිතය - (Sub Programs)

ක්‍රමලේඛයක ක්‍රියාවලි ප්‍රමාණය වැඩිවීමත් සමගම එය සංකීර්ණ වන නිසා කියවා තේරුම් ගැනීම, ක්‍රමලේඛය නඩත්තු කිරීම වැනි කටයුතු පහසු නොවේ. එබැවින් ක්‍රමලේඛ ලිවීමේදී ප්‍රධාන ක්‍රමලේඛය හැකිතාක් දුරට උප ක්‍රමලේඛ ලෙස කොටස් වශයෙන් ලියා දැක්වීම සාර්ථක ක්‍රමයක් ලෙස සැලකේ.

ප්‍රධාන ක්‍රමලේඛය හා සම්බන්ධ උප ක්‍රමලේඛ මගින් ප්‍රධාන ක්‍රමලේඛය වෙත ප්‍රතිදානයක් ආපසු ලබා දීමට හැකි සහ නොහැකි වශයෙන් උප ක්‍රමලේඛ වර්ග දෙකකි. ප්‍රතිදානයක් ආපසු ලබා දිය හැකි උප ක්‍රමලේඛ ශ්‍රිත හෙවත් Functions ලෙස ද, ප්‍රතිදානයක් ආපසු ලබා දිය නොහැකි එහෙත් යම් කාර්යයක් ඉටුකරන උප ක්‍රමලේඛ කාර්ය පටිපාටිය හෙවත් Procedures ලෙස ද හැඳින්වේ.

❖ උප ක්‍රමලේඛ හඳුන්වාදීම.

ප්‍රධාන ක්‍රමලේඛය ආරම්භ කිරීමට ප්‍රථමයෙන් ශ්‍රිත සහ කාර්ය පටිපාටි හඳුන්වා දිය යුතුය. ප්‍රධාන ක්‍රමලේඛය තුලදී උප ක්‍රමලේඛ වලට ඇමතිය (Calling the function or Procedure) යුතුය.



- කාර්ය පටිපාටියක් හඳුන්වා දීම සඳහා නිවැරදි කාරක රීතිය - (Syntax)
  Procedure Name_of_Procedure(Var name_of_variable : data type) ;

  උදා:- වෘත්තයක පරිධිය සෙවීම සඳහා කාර්ය පටිපාටිය
  Procedure calculate_Circumferance (var radius : real);

- ශ්‍රිතයක් හඳුන්වා දීම සඳහා නිවැරදි කාරක රීතිය - (Syntax)
  Function Name_of_Function(Var name_of_variable : data type) : data type ;

  උදා:- වෘත්තයක වර්ගඵලය සෙවීම සඳහා ශ්‍රිතය
  Function calculate_Area(var radius : real): real ;

උදා. 24: වෘත්තයක වර්ගඵලය සහ පරිධිය ගණනය කිරීම සඳහා කාර්ය පටිපාටි යොදා ගෙන ක්‍රමලේඛය ගොඩනැගීම.

```
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);
    readln;
end.
```

- **මෙම ක්‍රමලේඛයෙහි වල සිදුවන ක්‍රියාවලිය පැහැදිලිව තේරුම් ගෙන එය ලියා දක්වන්න.**

ක්‍රියාවලිය :- ....

උදා. 25: වෘත්තයක වර්ගඵලය සහ පරිධිය ගණනය කිරීම සඳහා ශ්‍රිත යොදා ගෙන ක්‍රමලේඛය ගොඩනැගීම.

```
program function_circle(input,output);
const pie=22/7;
var radius:real;
function processArea(var radius:real):real;
        var area:real;
        begin
            area:=pie*radius*radius;
            processArea:=area;
        end;
function processCircumferance(var radius:real):real;
        var circum:real;
```

```
        begin
           circum:=2*pie*radius;
           processCircumferance:=circum;
        end;
    begin
       writeln('Enter Radius');
       read(radius);
       writeln('Circumferance = ',processCircumferance(radius));
       writeln('Area = ', processArea(radius));
       readln;
    end.
```

- **මෙම ක්‍රමලේඛයෙහි වල සිදුවන ක්‍රියාවලිය පැහැදිලිව තේරුම් ගෙන එම ක්‍රියාවලිය සහ ප්‍රතිදානය ලියා දක්වන්න.**

ක්‍රියාවලිය :-

....
....
....
....
....
....
....
....
....
....
....
....

ප්‍රතිදානය:-

....
....
....
....

උදා. 26: **මෙම ක්‍රමලේඛයෙහි** getData , process_mult යන උපක්‍රමලේඛ දෙක කාර්යය පටිපාටි (procedure) භාවිතයෙන් ලියා ඇති අතර process_sum ශ්‍රිත (function) භාවිතයෙන් ලියා ඇත. මෙම ක්‍රමලේඛයෙහි වල සිදුවන ක්‍රියාවලිය පැහැදිලිව තේරුම් ගෙන එම ක්‍රියාවලිය සහ ප්‍රතිදානය ලියා දක්වන්න.

```
program procedure_test (input,output);
var  num1, num2,summation : integer;
procedure getData(var m1,m2 : integer);
        begin
              writeln ('Enter Number1');
              read(m1);
              writeln ('Enter Number2');
              read(m2);
        end;
```

```
procedure process_mult (var x1, x2 : integer);
        var ans : integer;
           begin
                 ans := x1 * x2;
                 writeln ('result = ',ans);
           end;
function process_sum (var p1, p2,sum : integer):integer;
           begin
                 sum := p1 + p2;
                 process_sum:=sum;
           end;
begin
           getData (num1, num2);
           process_mult (num1, num2);
           writeln('sum of two numbers is',process_sum(num1,num2,summation));
end.
```

ක්‍රියාවලිය :-

....

....

....

....

....

....

....

....

....

....

....

....

ප්‍රතිදානය:-

....

....

....

....

*****