PROGRAMMING

(Part-II)

පැස්කල් ක්‍රමලේඛ භාෂාව

හඳුන්වන (Identifiers)

- ඇවුරුණු පද යොදාගත නොහැක.
- ඉංග්‍රීසි හෝඩියේ අක්ෂරයකින් ආරම්භ කළ යුතුය.
- පළමු අක්ෂරයට පසුව ඉලක්කම් හෝ (_) භාවිත කළ හැක.
- අක්ෂර භේදය බල නොපායි.

ඇවුරුණු පද (Reserved words)

පැස්කල් භාෂාවේ භාවිත කරන ඇවුරුණු පද පැස්කල් භාෂාව හඳුන්වා දීම සඳහා භාවිතා කෙරේ. එබැවින් එවා හදුන්වන සඳහා භාවිත නොකෙරේ.

පැස්කල් භාෂාවේ සම්මත දත්ත ප්‍රරූප

- **Integer**- ධන හෝ ඍණ නිඛිල සංඛ්‍යා
- **Real** - ධන හෝ ඍණ දශම සංඛ්‍යා
- **Boolean**
- **Char** - යතුරුපුවරුවේ ඇති ඕනෑම අනුලක්ෂණයක්
- **String** - ඕනෑම අනුලක්ෂණ අනුක්‍රමයක්

ClassWork.LK

<u>ක්‍රමලේඛනයේදී යොදාගන්නා විචල්‍ය හා නියත</u>

<u>විචල්‍ය (Variables)</u>

ක්‍රමලේඛයක් ක්‍රියාත්මක වන විට හඳුන්වන සඳහා පවරන ලද අගයන් වෙනස් වන හඳුන්වන විචල්‍ය ලෙස හැඳින්වේ.

ඇවුරුණු පදය - Var

<u>නියත (Const)</u>

ක්‍රමලේඛයක් ක්‍රියාත්මක වන විට හඳුන්වන සඳහා පවරන ලද අගයන් වෙනස් නොවන හඳුන්වන නියත ලෙස හැඳින්වේ.

ඇවුරුණු පදය - Const

ClassWork.LK

කාරක (Operaters)

අංක ගණිතමය කාරක

එකතු කිරීම → +

අඩු කිරීම → -

ගුණ කිරීම → ×

බෙදීම → /

පූර්ණ සංඛ්‍යාමය බෙදීම→ DIV

බෙදීමෙන් පසු ශේෂය → MOD

සැසඳුම් කාරක

විශාල → >

විශාල හෝ සමාන → >=

කුඩා → <

කුඩා හෝ සමාන → <=

සමාන → =

අසමාන → <>

ClassWork.LK

තාර්කික කාරක

AND කාරකය

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

OR කාරකය

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

ClassWork.LK

NOT කාරකය

ප්‍රකාශනය	NOT (ප්‍රකාශනය)
අසත්‍ය	සත්‍ය
සත්‍ය	අසත්‍ය

කාරක ප්‍රමුඛතාව

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

ClassWork.LK

පැස්කල් ක්‍රමලේඛයක අඩංගු මූලික කොටස්

program addNum (input,output); ක්‍රමලේඛයේ නම

Var num1,num2,total : integer; ← විචල්‍ය හඳුන්වා දීම

Avg: real;

Begin ← ප්‍රධාන ක්‍රමලේඛයේ ආරම්භය

Write(‘Welcome Pascal Programming’):

විවරණ දැක්වීම → {calculate total and average of two numbers}

Writeln(‘Enter First Number’);

Read(num1); ← ආදාන

Writeln(‘Enter second Number’);

Read(num2);

ප්‍රතිදාන

Total :- num1 + num2; ← ක්‍රියාවලිය දැක්වෙන ප්‍රකාශ

Avg :- total/2:

Writeln(‘total is’ , total);

Writeln(“average is” .avg);

End ← ක්‍රමලේඛයේ අවසානය

- විවරණ දැක්වීමට (*..........*) භාවිතා කෙරේ.
- ප්‍රකාශනයක් අවසානයේ ලකුණ යෙදිය යුතුය.

ClassWork.LK

වරණ පාලන ව්‍යූහය සම්බන්ධ වන ක්‍රමලේඛ නිර්මාණය

සරල ප්‍රකාශය

IF… THEN… ENDIF

මෙහිදී කොන්දේසිය තෘප්ත නම් පමණක් ප්‍රකාශනය ක්‍රියාත්මක වේ.

IF… THEN.. ELSE… ENDIF

මෙහිදී කොන්දේසිය තෘප්ත නම් ප්‍රකාශනය (1) ද තෘප්ත නොවේනම් ප්‍රකාශනය (2) ද ක්‍රියාත්මක වේ.

NESTED IF

මෙහිදී කොන්දේසියකට පසුව තවත් කොන්දේසියක් වශයෙන් කොන්දේසි සමූහයක් යෙදෙන අවස්ථාවක් භාවිතා කෙරේ.

ClassWork.LK

පුනර්කරණ භාවිතයෙන් ක්‍රමලේඛ ලිවීම

FOR-DO ව්‍යූහය

පුනර්කරණ වාර ගණන නිශ්චිතව දන්නා අවස්ථා වලදී භාවිතා වේ.

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.
```

WHILE-DO ව්‍යූහය

පුනර්කරණ වාර ගණන නිශ්චිතව නොදන්නා අවස්ථා වලදී භාවිතා වේ.

ClassWork.LK

උදා:

```
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 වඩා අඩු වන විට පුනර්කරණය සිදු වේ.
- number විචල්‍යයේ අගය 11 වු විට පුනර්කරණය නවතී.

Repeat Until ව්‍යූහය

උදා : පැස්කල් යන වචනය පුනර්කරණය වීම.

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

ClassWork.LK

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

<u>නීඩිත පාලන ව්‍යූහවල (Nested Control Structures) අවශ්‍යතාව</u>

ගැටලුවක ස්වභාවය මත වරණයකින් පසුව පුනර්කරණයක් හෝ පුනර්කරණයකින් පසුව වරණයක් යෙදිය හැක. මෙවැනි අවස්ථාවල නීඩිත පාලන ව්‍යූහ යොදාගැනීමට සිදු වේ.

ClassWork.LK

වරණය තුළ පුනර්කරණ යොදා ගැනීම

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.
```

ClassWork.LK

පුනර්කරණය තුළ වරණය යොදා ගැනීම

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.
```

ClassWork.LK

අරාව (Array) භාවිතය

අරාවන් භාවිතයේ අවශ්‍යතාව

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

ඒකමාන අරාවක් අර්ථ දැක්වීම

අනුක්‍රමික ව නිශ්චිත ප්‍රමාණයක එක ම ප්‍රරූපයකට අයත් දත්ත තැන්පත් කිරීම සඳහා භාවිත කරන දත්ත ව්‍යුහයකි.

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

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

ClassWork.LK

අරාවක ලක්ෂණ

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

අරාවකට අගයන් පැවරීම (Assign values to an array)

අරාවක අවයවවලට වෙන වෙන ම අගයයන් පවරනු ලැබේ.

ClassWork.LK

<u>අරාවක අගයන් ප්‍රකාශයට පත් කිරීම (Display values in an array)</u>

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
```

ClassWork.LK

```
            if marks[i] > max then max := marks[i];
        writeln('Maximum marks = ', max);
        writeln('Average marks = ',avg);
end
```

<u>උපක්‍රමලේඛ භාවිතය</u>

<u>උපක්‍රමලේඛ වර්ග</u>

- ප්‍රතිදානයක් ආපසු ලබා දිය හැකි උපක්‍රමලේඛ **ශ්‍රිත** (Function) ලෙස ද,
- ප්‍රතිදානයක් ආපසු ලබා දිය නොහැකි උපක්‍රමලේඛ **කාර්ය පටිපාටිය** (Procedure) ලෙස ද හැඳින්වේ.

<u>උපක්‍රමලේඛ හඳුන්වා දීම</u>

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.

ClassWork.LK

කාර්ය පටිපාටියක් හඳුන්වා දීම සඳහා නිවැරදි කාරක රීතිය:

Procedure Name_of_Procedure (name_of_variable: data type);

ශ්‍රිතයක් හඳුන්වා දීම සඳහා නිවැරදි කාරක රීතිය:

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

ClassWork.LK

ClassWork.LK