Mo Tu We Th Fr Sa Su

Date / /

APPSNA BU

Pseudo code - Formal representation of programming logic in a language independent manner.

```
start
  Declarations
    :
  statements        } structure
    :
stop
```

Ex

```
start
    n ← 0(integer);
    m ← 0(integer);
    s ← 0(integer);
  Accept n, m;
    s ← n + m;
    display s;
stop
```

→ Each variable declared in pseudo code is initialized with some value or null for that data type →
- Integer - 0
- Double - 0.0
- Char - \0
- String - ""

} default values

Ex `age ← 0(integer);`
`Name ← "John"(string);`

→ Each statement is terminated with a semi-colon

Rules → Use good naming style for the variables declared
→ Identation is mandatory & improves readability.
→ Test your code using Dry Run technique.

Dry Run

Step No	n	m	s	o/p
1.	0			
2.		0		
3.			0	
4.	45	65		
5.			110	
6.				110

Mo Tu We Th Fr Sa Su

Date / /

Ex 1 PC for accept principle, rate of interest & time. Calculate simple interest

Start

principleAmount ← 0.0 (float);
rateofInterest ← 0.0 (float);
timeINYears ← 0 (integer);
simpleInterest ← 0.0 (float);
Accept principleAmount, rateofInterest, timeinYears;
simpleInterest ⇐ (principleAmount * rateOfInterest * timeInYears) / 100.0;

Display simpleInterest

stop

2) PC to accept 2 numbers, Display them, swap two numbers & display them again.

start

num1 ← 0 (integer);
num2 ← 0 (integer);
temp ← 0 (integer);
Accept num1, num2;
Display num1, num2;
temp ← num1;
num1 ← num2;
num2 ← temp;
Display num1, num2;

stop

Mo Tu We Th Fr Sa Su

Date / /

Conditional statements

format → if (condition) then
statement 1;
statement 2;
else
S1;
S2;
:
end if;

Ex → if (Withdraws < balance) then
balance ← balance - Withdraws;
else
display "insufficient funds";
end if;

Relational operators → >, <, >=, <=, !=, ==
Logical Operators → and, or, Not
Arithmetic " → +, -, *, /, %, ^

3) PC to accpt a num & display whether it is an even (or) odd

Start
num ← 0(integer);
Accept num;
if (num % 2 == 0) then
Display "Even number";
else
Display num "is odd number";
end if;
stop

Mo Tu We Th Fr Sa Su

Date / /

4) PC to accept a double value, seperate the whole value from the fractional value & store them in 2 variables, Display them

start

value ← 0.0(double);

Accept value;

wholeValue ← 0 (Integer);

fractionalValue ← 0.0 (Double);

wholeValue ← INT(value);

fractionalValue ← value − wholeValue;

Display wholeValue, fractionalValue;

stop

Nested if (multiple branching)

```
if(condition) then
    statement 1;
    :
    if (condition) then
    statement 2;
    else
    statement 3;
    endif;
else
    S1;
    if(con) then
    S2;
    else
    S3;
    endif;
    :
endif;
```

Ex-6 PC to find largest & second largest of 3 nums

start

num1 ← 0 (integer);
num2 ← 0 //
num3 ← 0 //
largest ← 0 //
secondlargest ← 0 //

Accept num1, num2, num3;
largest ← num1;
secondlargest ← num2;
if(num3 > largest) then
secondlargest ← largest;
largest ← num3;
else if (num3 > secondlargest) then
secondlargest = num3
endif

stop

Mo Tu We Th Fr Sa Su

Date / /

5) Display Avg & Total PC

Start

studentName ← "" (string);

score1 ← 0 (integer); S → S2, S3

Accept studentName, score1, S2, S3;

total ← 0 (integer);

avg ← 0.0 (float);

total ← S1 + S2 + S3;

avg ← total / 3.0;

studentClass ← "" (string);

if (S1 >= 60 AND S2 >= 60 and S3 >= 60) then

studentClass ← "1st class";

else if (S1 >= 50 and S2 >= 50 and S3 >= 50) then

studentclass ← "2nd class";

else if (S1 >= 35 and S2 >= 35 and S3 >= 35) then

studentclass ← "3rd class";

else

studentclass ← "failed";

endif;

Display studentName, total, avg, studentclass;

End

stop

Tu We Th Fr Sa Su

Date / /

⇒ PC for compute annual salary.

start

name ← "" (String);
empID ← "" (string);
basic ← 0.0(float);
specialAllowance ← 0.0(float);
bonusPercentage ← 0.0(float);
taxSavingInvest ← 0.0(float);
grossMonthlySalary ← 0.0(float);
grossAnnualSalary ← 0.0(float);
annualTaxableIncome ← 0.0(float);
annualgrosssalary ← 0.0(float);
annualNetsalary ← 0.0(float);
taxpayable ← 0.0(float);
Accept name, empID, basic, specialAllowance, bonuspercentage, taxSavingInvestment;
grossMonthlySalary ← basic + specialAllowance;
grossAnnualSalary ← grossMonthlySalary * 12 + (bonusPercentage /100) * grossMonthlySalary);
annualTaxableIncome = grossAnnualSalary - taxSavingInvestm
if (annualTaxableIncome < 100000) then
annualTaxableIncome = 0;
else if (annualTaxableIncome < 150000) then
annualTaxableIncome = annTI - 100000;
" = AnnualTaxIncom * 0.2;
else

Mo Tu We Th Fr Sa Su Date / /

annualTaxableIncome = aTI − 150000 ;

" = (50000 * 0.2) + (annualTI * 0.3) ;

endif ;

annualGrossSalary = grossAnnualSalary ;

annualNetSalary = annualGrossSalary − annualTaxableIncome ;

taxPayable = annualTaxableIncome ;

Display name, empID, annualgrossSalary, annualNetSalary, Taxpayable ;

Stop

Structured loops → while, Do-while (Repeat ....Until), for

while(conditions)
statements ;
end while ;

Ex
i ← 1 ;
while (i <= 10)
display i ;
i ← i+1 ;
endwhile ;

8) Start sum of odd numbers till N

n ← d(integer) ;
sum ← d(integer) ; i ← 1(integer) ;
Accept n ;
for (i ≤ n)
sum = sum + i ;
i = i+2 ;
end for
display sum ;
stop

StepNo	N	Sum	i	o/p
1.	0	0	1	
2.	3			
3.		4	3	
4.				4

Mo Tu We Th Fr Sa Su

Date / /

9) Reverse a Num

```
Start
    num ← 0 (integer);
    reversenum ← 0(integer);
    Accept num;
    while(num>0)
    reversenum=(reversenum*10)+(num%.10);
    num = num/10 ;
    End while
    Display reversenum;
Stop
```

SNO	num	Rnum	O/P
1.	0	0	
2.	548		
3.		4	
4.	5		
5.		45	
6.			45

10) Display num in words

```
Start
    num ← 0(integer) ;
    dNames[10] ← ""(string);
    res ← "" (string); digit ← 0(integer);
    dNames[0] ← "zero";
    dNames[1] ← "One";
    dNames[9] ← "Nine";
    Accept num ;
    while (num>0)
    digit ← num % 10 ;
    result ← digit dNames[digit] + " " + result ;
    num ← num/10;
    End while
    Display result;
Stop
```

Mo Tu We Th Fr Sa Su

Date / /

11) factorial

start

num ← 0 (integer);

fact ← 1 (integer);

Accept num;

if (num<0) then

Display "Factorial of -ve num is not possible";

else if (num==0) then

Display "factorial of 0 is 1";

else

for i←1 to num

fact = fact * i;

end for

Display "factorial of ", num, "is", fact;

End if

stop

12) Display decimal num in binary form

start

decimal ← 0 (integer);

binary ← "" (string);

Accept decimal;

while (decimal >0)

remainder ← decimal % 2;

binary = remainder + binary;

decimal = decimal / 2;

End while

Display binary;

stop

Mo Tu We Th Fr Sa Su

Date / /

13) Binary to decimal

start

binary ← "" (string);

decimal ← 0 (integer);

Accept binary;

for i ← length(binary) to 1

bit ← Substring(binary, i, 1)

decimal ← decimal + (INTEGER(bit) * (2 ^ (length(binary) - i)));

end for

Display decimal;

stop

14-1) 2^{*2} 4^{*4} 6^{*6} 8^{*8}
4, 16, 36, 64 .... N

start

n ← 0(integer);

i ← 0(integer);

Accept n;

For i ← 2 to n [step 2]

t ← 0(integer);

t ← i * i

display t;

end for

stop

2) 1, -2, 3, -4, 5, -6 .... N

start

n ← 0(integer);

i ← 0(integer);

Accept n;

for i ← 1 to n

t ← 0(integer

if (i % 2 == 0) then

t = -i;

else

t = i;

end if

display t;

end for

stop

Mo Tu We Th Fr Sa Su

Date / /

2) 1, 4, 27, 256, 3125 ... N

(1, 2, 3, 4, 5)

```
Start
  n ← 0(integer);
  i ← 0(integer);
  Accept n;
  for i ← 1 to N
    t ← 0(integer);
    t ← i ^ i;
    display t;
  end for
Stop
```

4) 1, 4, 7, 12, 23, 42, 77 ... N

(+11, 19, 35; 3, 3, 5, 11, 19, 35)

```
Start
  n ← 0(integer);
  Accept n;
  Declare Array a[n] integer;
  a[0] ← 1(integer); a[1] ← 4(integer);
  a[2] ← 7(integer);
  for i ← 3 to N
    a[i] ← a[i-1] + a[i-2] + a[i-3];
  end for
  display a;
Stop
```

5) 1, 4, 9, 25, 36, 49, 81, 100 ... N

($1^2, 2^2, 3^2, 5^2, 6^2$)

```
Start
  n ← 0(integer);
  i ← 0(integer);
  Accept n;
  for i ← 1 to n:
    if(i != 4) then
      t ← i ^ 2;
      display t;
    end if
  end for
Stop
```

6) 1, 5, 13, 29, 49, 77 ... N

(4, 8, 8, 16, 4, 20, 28; 4, 8)

```
Start
  n ← 0(integer);
  i ← 0(integer);
  Accept n; t ← 1 (integer);
  for i ← 1 to n
    if (i % 2 != 0) then
      t = t + 4;
    else
      t = t + 8;
    display t;
  end for
Stop
```

Mo Tu We Th Fr Sa Su

Date / /

15) Sum of Prime Numbers

```
start
    n ← 0(integer);
    m ← 0(integer);
    sum ← 0(integer);
    Accept n, m;
    for i ← n to m
        isPrime ← 'true' (boolean);
        if (num < 2) then
            isPrime = false;
        else if num == 2 then
            isPrime = True;
        else if num % 2 == 0 then
            isPrime = false;
        else
            divisor ← 0(integer);
            for divisor ← 3 to sqrt(num) [step 2]
                if (num % divisor == 0) then
                    isPrime = false
                    break
                end if
            end for
        end if
        if isPrime then
            sum = sum + num
            display num
        end if
    end for
    display sum;
    stop
```

Mo Tu We Th Fr Sa Su

Date / /

for loop

```
for i 1 to 10
display i;
end for
```

Nested for

```
for i<-1 to 10
    for j<-1 to 10
        display (i+j);
    end for
end for
```

16-1) 1, -2, 6, -15, 31, -56 .... N

```
start
    n<-0(integer);
    i<-0(integer);
    t<-1(integer);
    for i<-1 to n
        display t;
        if (i%2==0) then
            t=t-(i*2+1);
        else
            t=t+(i*2+1);
        end if
    end for
stop
```

2) 1, 1, 2, 3, 5, 8, 13 .... N

```
start
    n<-0(integer);
    i<-0(integer);
    a,b<-1(integer);
    Accept n;
    for i<-1 to n-1
        t<-a+b;
        display t;
        a<-b;
        b<-t;
    end for
stop
```

3) $1, -2^2, 4, -6^2, 7, -10^2, 10, -14^2 \cdots N$

-3 6 -9

Mo Tu We Th Fr Sa Su Date / /

4) 1, 5, 8, 14, 27, 49 ....

17) Power of x to the N

Start
  x ← 0.0(float);
  n ← 0(integer);
  res ← 0.0 (float);
  Accept x, n;
  res ← 1;
  for i ← 1 to n
    res ← res * x
  end for
  display res;
Stop

18) Reverse a string

Start
  input ← "" (String);
  reverse ← "" (string);
  Accept input;
  for i ← length(input) to 1
    reverse = reverse + SUBSTRING
      (input, i, 1)
  end for
  display reverse;
Stop

(or)

```
StringBuilder reverse = new StringBuilder();
for( int i = input.length()-1;
      i >= 0; i--) {
  reverse.append (input.charAt(i));
}
```

19) Check for palindrome

Start
  reverse string code
  if (input == reverse) then
  display " Palindrome ";
  else
    display " not Palindrome";
  end if
Stop

Mo Tu We Th Fr Sa Su

Date / /

20)-1) * * * * *
* * * * *
* * * * *

```
Start
  n ← 0(integer);
  Accept n;
  for i ← 1 to n
    for j ← 1 to 5
      display "*";
    end for
  end for
Stop
```

2)
1 1 1 1 1
2 2 2 2 2
3 3 3 3 3

```
Start
  n ← 0(integer);
  Accept n;
  for i ← 1 to n
    current = i;
    for j ← 1 to 5
      display current;
    end for
      display \n;
  end for
Stop
```

3) 1 2 3 4 5
1 2 3 4 5
1 2 3 4 5

```
start
  rows ← 0 (integer);
  cols ← 0(integer);
  Accept rows, cols;
  for i ← 1 to rows
    for j ← 1 to cols
      display j;
      if (j < cols) then
      display " ";
      end if
    end for
    display "\n";
end for
stop
```

4)
*
* *
* * *
* * * *

```
start
  n ← 0(integer);
  Accept n;
  for i ← 1 to n;
    for j ← 1 to i;
      display "*";
    end for
    display "\n";
  end for
stop
```

Mo Tu We Th Fr Sa Su

Date / /

5)
```
1
1 2
1 2 3
1 2 3 4
```

```
Start
    n ← 0(integer);
    Accept n;
    for i ← 1 to n
        for j ← 1 to i
            display j;
        end for
        display "\n";
    end for
Stop
```

6)
```
1
2 3
4 5 6
7 8 9 10
```

```
start
    n ← 0 (integer);
    current ← 1 (integer);
    for i ← 1 to n
        for j ← 1 to i
            display current;
            current = current + 1;
            display " ";
        end for
        display "\n";
    end for
Stop
```

7)
```
1
1 2
3 5 8
```
} Fibonacci

```
start
    n ← 0(integer);
    Accept n;
    prev ← 0(integer);
    curr ← 1(integer);
    for i ← 1 to n
        rowstart ← prev + 1
        for j ← 1 to i
            print rowstart;
            rowstart ← rowstart + curr;
            prev ← curr;
            curr ← rowstart;
            if (j < i) then
                display " ";
            end if
        end for
        display "\n";
    end for
stop
```

Mo Tu We Th Fr Sa Su

Date / /

8) 1
-4 9
-16 25 -36

```
start
  n←0(integer);
  Accept n;
  for i←1 to n
    curr←0(integer);
    if (i%2==1) then
      curr←i*i;
    else
      curr←-(i*i);
    end if
    for j from 1 to i
      display curr
      if (i%2==1) then
        curr = curr+1
      else
        curr = curr-1
      end if
      if j<i then
        display " ";
      end if
    end for
    display "\n";
  end for
stop
```

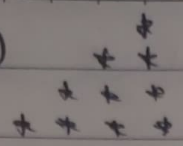
9) 1
1 2
6 24 120 } factorial

```
Start
  n←0(integer);
  Accept n;
  for i←1 to n
    curr←1(integer);
    fact←1(integer);
    for j←1 to i
      display fact;
      curr = curr+1;
      fact = fact * curr;
      if (j<i) then
        print " ";
      end if
    end for
    display "\n";
  end for
stop
```

Mo Tu We Th Fr Sa Su

Date / /

10)



Start

n ← 0(integer);

Accept n;

for i ← 1 to n

for j ← 1 to n-i

display " ";

end for

for k ← 1 to i

display "*";

end for

display "\n";

end for

stop

11)

Start

rows ← 0(integer);

Accept rows;

for i ← 0 to rows

for j ← 0 to 2*(rows-i)

display " ";

end for

for k ← 0 to 2*i+1

display "*";

end for

display "\n";

end for

stop

Arrays - collection of elements of same datatype (Homogenous)

Declare Array arrayname Of size data-type;

Ex Declare Array a of 10 integer;

2-D Array

Declare Array arrayname of [size, size] datatype;

Ex Declare Array a of [10, 10] integer;

Mo Tu We Th Fr Sa Su

Date / /

21) Linear search

```
start
  n ← 0(integer);
  searchValue ← 0(integer);
  isfound ← "false"(boolean);
  Accept n, searchValue;
  Declare Array a of [n] integer;
  for i ← 0 to n-1
    Accept a[i];
  end for
  for i ← 0 to n-1
    if (a[i] == searchValue) then
      isfound ← true;
    end if
  end for
  if isfound
    display "found";
  else
    display "not found";
  end if
stop
```

22) Binary search

same ↓

```
  l ← 0(integer);
  r ← a.length - 1;
  while (l <= r)
    m ← l + (r-l) /2;
    if (a[m] == searchValue)
      isfound ← true;
    end if
    if (a[m] < searchValue)
      l ← m + 1;
    else
      r ← m - 1;
    end if
  end while
  if isfound
    display "found";
```

Mo Tu We Th Fr Sa Su

Date / /

23) Transpose of m*n matrix

```
Start
  m←0(integer);
  n←0(integer);
  Accept m, n;
  Declare Array a of [m,n] integer;
  for i←0 to m-1
    for j←0 to n-1
      Accept a[i][j];
    end for
  end for
  for j←0 to n-1
    for i←0 to m-1
      Display a[i][j];
      if (i<m-1) then
        display " ";
      end if
    end for
    display "\n";
  end for
stop
```

24) Check if matrix is Identity Matrix

```
n * n
isidentity ← true(boolean);
for i←0 to n-1
  for j←0 to n-1
    if (i=j AND a[i][j]!=1) then
      isidentity ← false
    else if(i≠j AND a[i][j]!=0)
      isidentity ← false
    end if
  end for
end for
if isidetity
  display "Identity";
else
  display "Not Identity";
end if
stop
```

25) symmetric or not

```
  issymmetric ← true(boolean);
  for i←0 to n-1
    for j←0 to n-1
      if (a[i][j] ≠ a[j][i]) then
        issymmetric ← false;
      end if
    end for
  end for
  if issymetric then
    display "symmetric";
  else
    display "Not symmetric";
stop
```