PPS :-

Page No. 1
Date

Decision Making & Looping

↦ Points :-
- Looping Statements
- Nesting of Control Structures
- Break & Continue Statements

↦ Looping :- (Defⁿ)

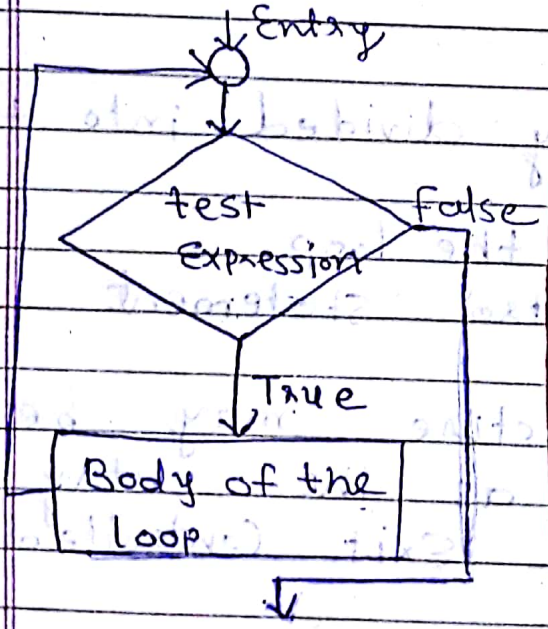
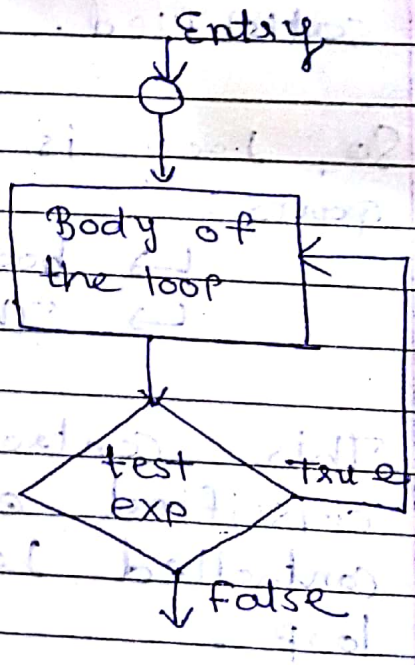
↦ In looping, a sequence of statements are executed until some conditions for termination of the loop are satisfied.

↦ So, loop is mainly divided into two parts :-
  - ↳ body of the loop
  - ↳ The control statement

↦ This control structure may be classified either as the entry controlled loop or Exit controlled loop.

Prepared by :-
Ms. Jinal Zala

→ Difference betⁿ

Entry Controlled Loop	Exit Controlled Loop
↦ In entry controlled loop, the control conditions are tested before the start of the loop execution. If conditions are not satisfied, then the body of the loop will not be executed.	↦ In an exit-controlled loop, the test is performed at the end of the body of the loop & therefore the body is executed unconditionally for the first time.
↦ 	
↦ Ex:- For, while loops are entry controlled loops.	↦ Ex:- do...while loop is exit controlled loop.

⇒ Looping Process include following steps:-

1. Initialization of Conditional variable.
2. Execution of the statements in the loop. (Body)
3. Test Expression or Termination Condition for loop.
4. Increment / Decrement or updating the conditional variable.

⇒ Types of Loop:-

1. The While statements - (Entry - Controlled)
2. The do statements - (Exit Controlled)
3. The for statements - (Entry - controlled)

1. The While statement:-

⇒ Syntax:-

```
Initialization;

while (test_exp)
{
    Body of loop;
    Increment / Decrement;
}

next_statement;
```

↦ Cone to display 1 to 10 nums using while loop.

```
void main()
{
    int i=1;          // Initialization
    while (i<=10)     // Termination Condition
    {
        pfc"%d\n",i); // Body of Loop
        i=i+1;        // Increment
    }
    getch();          // next- st;
}
```

2. The do statement :-

Syntax :-

```
Initialization ;
do
{
    Body of the loop;
    Increment / Decrement;
}
while (test- exp);
next - st;
```

H WAP to display 1 to 10 nums using do...while loop.

```
void main()
{
    int i = 1;      // Initialization
    do
    {
        printf("%d\n", i);   // Body of the loop
        i++;                 // Increment
    }
    while(i<=10);    // Termination Condition
    getch();         // next st
}
```

3. The for Statement:-

Syntax:-

```
for (initialization; test_condition; inc/dec)
{
    Body of the loop;
}
next statement;
```

↳ WAP to display 1 to 10 nums using for loop.

```
void main()
{
    int i;
for(i=1; i<=10; i++)
{
    printf("%d\n", i); // Body of the loop
}
getch(); // next St
}
```

Initialization
condition
Increment

★ Examples:-

1. WAP to display "Hello" five times.

```
void main()
{
    int i=1;

    while(i<=5)
    {
        printf("Hello\n");
        i=i+1;
    }
    getch();
}
```

2. WAP to display 20-30 nums.

```
Void main()
{
    int i;

    for(i=20; i<=30; i++)
    {
        Pfc"%d\n", i);
    }
    getch();
}
```

3. WAP to display multiplication table.

Ex:- n = 5 then print 5 * 1 = 5
5 * 2 = 10
⋮
5 * 10 = 50

```
Void main()
{   int i=1, n;
    Pfc"Enter n:");
    Sfc"%d", &n);

    While (i<=10)
    {
        Pfc"%d * %d = %d\n", n, i, n*i);
        i++;
    }
    getch();
}
```

4. WAP to display sum of 1 to 100.

1.) while loop:-

```
Void main()
{
    int i=1, sum=0;
    while (i<=100)
    {
        sum = sum+i;
        i++;
    }
    Pfc" Sum of 1 to 100 = %d", sum);
    getch();
}
```

2.) do... while loop:-

```
void main()
{
    int i=1, sum=0;
    do
    {
        sum = sum+i;
        i++;
    }
    while (i<=100);
    Pfc" Sum of 1 to 100 = %d", sum);
    getch();
}
```

H 3) for loop:-

```
Void main()
{
    int i, sum=0;
    for(i=1; i<=100; i++)
    {
        sum = sum+i;
    }
    pfc" Sum of 1 to 100 = %d", sum);
    getch();
}
```

5. WAP in C for finding sum of 1 to k. The number k should be read from the keyboard using scanf().

H

```
Void main()
{
    int i=1, sum=0, k;
    pfc"Enter the limit of k:");
    Sfc"%d", &k);
    while(i<=k)
    {
        sum = sum+i;
        i++;
    }
    pfc" Sum of 1 to %d = %d", k, sum);
    getch();
}
```

6. CAP to print multiple of N from given range of unsigned integers.
For example:-
if N=5 & range is [17,45] it print:
20, 25, 30, 35, 40, 45,

```
Void main()
{
    int start, stop, N, i;
    pfc" Enter your range :");
    sfc"%d %d", &start, &stop);
    pfc" Enter N :");
    sfc"%d", &N);

  for(i= start; i<=stop; i++)
  {
      if(i%N==0)
      pfc"%d\t", i);
  }
  getch();
}
```

7. CAP to find sum of all integers greater than 100 & less than 200 and are divisible by 5.

```
Void main()
{
    int i=101, sum=0;
    do
    {
```

```
    if (i%5 ==0)
    sum = sum + i;
    i++;
  }
  while (i<200);
  pfc("Sum = %d", sum);
  getch();
}
```

8. WAP to count ODD & EVEN numbers from given 10 numbers.

```
Void main()
{  int i, odd=0, even=0, n;

   for(i=1; i<=10; i++)
   {
     pfc("Enter no %d", i);
     sfc("%d", &n);
     if(n%2!=0)
     odd++;
     else
      even++;
   }
pfc("Total odd nums = %d", odd);
pfc("\n Total even nums = %d", even);
getch();
}
```

→ O/P:- 2 5 7 9 2 3 4 7 8 55
Total odd nums = 6
Total even nums = 4

9. WAP to find the sum of first N odd nums.

```
Void main()
{
    int i, sum=0, n, i;
Pfc"Enter n:"); Sfc"%d", &n);
for(i=1; i<=(2*n); i++)
{
    if (i%2!=0)
    sum = sum+i;
}
Pfc" Sum of first N odd =%d", sum);
getch();
}
```

10. WAP to print 1+1/2+1/3+...+1/N series.

```
Void main()
{   int i, n;
    Pf c"Enter the limit of N:");
    Sf c"%d", &n);

    for(i=1; i<=n; i++)
    {
        if (i==1)
        Pf c"%dt", i);
        else if (i==n)
        Pf c"1/%d", i);
        else
        Pfc"1/%dt", i);
    }
```

11. WAP to find factorial of a number.

↦ Method 1:-

```
Void main()
{
    int n, fact = 1, s;
    pfc"Enter n:");
    Sfc"%d", &n); s=n;
    while (n >= 1)
    {
        fact = fact * n;
        n--;
    }
    pfc" factorial of %d = %d", s, fact);
    getch();
}
```

↦ Method 2:-

```
Void main()
{   int i, n, fact=1;
    pfc "Enter n:");
    Sfc"%d", &n);
    for(i=1; i<=n; i++)
    {
        fact = fact * i;
    }
    pfc" fact ans = %d", fact);
    getch();
}
```

12. WAP to generate fibonacci series of n numbers.

Ex:- n = 6 → 1 1 2 3 5 8
↑ 6 digits in series.

```
Void main()
{
    int i, a=1, b=1, c, n;

    Pfc"Enter n:");
    Sfc"%d", &n);
    Pfc"%4d %4d", a, b);

    for(i=1; i<=(n-2); i++)
    {
        c = a+b;
        Pfc"%4d", c);
        a=b;
        b=c;
    }
    getch();
}
```

13. WAP to reverse a number. (Ex:- 123 → 321)

```
Void main()
{
    int n, rev=0;
    Pfc"Enter n:");
    Sfc"%d", &n);
    while(n!=0)
    { rev = rev*10;
      rev = rev + (n%10);
      n = n/10;
```

```
pfc"Reverse num = %d", rev);
getch();
}
```

14. WAP to check whether a given num is palindrome or not.
Ex:- (121 $\xrightarrow{\text{Reverse}}$ 121) - Palindrome
(123 $\xrightarrow{\text{Rev}}$ 321) - Not Palindrome

↦
```
{   int n, rev=0, s;
    pfc"Enter number:";
    Sfc"%d", &n);
    s=n;
    while (n!=0)
    {
        rev = rev * 10;
        rev = rev + (n%10);
        n = n/10;
    }

    if (s==rev)
    pfc"Given num is Palindrome");
    else
    pfc"Given num is not Palindrome");
    getch();

}
```

15. WAP to calculate sum of digits.
(Ex:- n=123 $\xrightarrow{\text{O/P}}$ 1+2+3 = (6))

```
Void main()
{
    int n, sum=0, digit;

    Pf("Enter n:");
    Sf("%d", &n);
    While(n!=0)
    { digit = n%10;
      sum = sum+digit;
      n = n/10;
    }
    pf("Sum of digits = %d", sum);
    getch();
}
```

16. WAP to find Out the given number is an armstrong number or not.
Ex:- num → 153 → $1^3+5^3+3^3$ = 153

```
Void main()
{ int num, digit, sum=0, temp;
  pf("Enter num:");
  sf("%d", &num);
  temp = num;

  While(num!=0)
  { digit = num%10;
    sum = sum+(digit*digit*digit);
    num = num/10;
  }
  if(sum==temp)
  Pf("Armstrong Num");
  else
```

```
    pfc" Not an Armstrong Num");
    getch();
}
```

17. WAP to print armstrong num betn 1 to 1000.

↦
```
Void main()
{
    int num, temp, digit, sum;
    pfc"Armstrong nums betn 1 to 1000 are:");
    for(num=1; num<=1000; num++)
    {
        temp = num;
        sum = 0;
        when(temp!=0)
        {
            digit = temp % 10;
            sum = sum + (digit * digit * digit);
            temp = temp / 10;
        }

        if(num == sum)
        pfc"%d\t", num);
    }
    getch();
}
```

↦ Output:-

Armstrong nums betn 1 to 1000 are
1 153 370 371 407

18. WAP to check whether a given number is Prime or not.

```
Void main()
{
    int num, j, flag=0;
    pf("Enter a num:");
    sf("%d", &num);

    for(j=2; j<=num/2; j++)
    {
        if(num%j==0)
        {
            flag=1;
            break;
        }
    }
    if(flag==0)
    pf("%d is a Prime", num);
    else
    pf("%d is not a Prime", num);
    getch();
}
```

19. WAP to find & Print prime nums betn the numbers 1 to n, where the number n should be read from the keyboard.

```
Void main()
{
    int i, num, flag, n;
```

```
pf("Enter n:");
sf("%d", &n);

for (num = 1; num <= n; num++)
{
    flag = 0;

    for (i = 2; i <= num/2; i++)
    {
        if (num % i == 0)
        {
            flag = 1;
            break;
        }
    }
    if (flag == 0)
    pf("\t%d", num);
}
getch();
}
```

↦ O/P:- Enter n - 30

2 3 5 7 11 13 17 19 23 29

20. WAP to find out sum of first & last digit of a given number.

↦

```
& void main()
{
    int num, digit, last, first, sum;
```

```
Pfc"Enter num.");
Sfc"%d", &num);
last = num %.10;
while (num != 0)
{
    digit = num %. 10;
    first = digit;
    num = num / 10;
}

sum = first + last;

Pfc"sum of first & last digit =
                        %d", sum);

getch();

}
```

↦ o/p:-

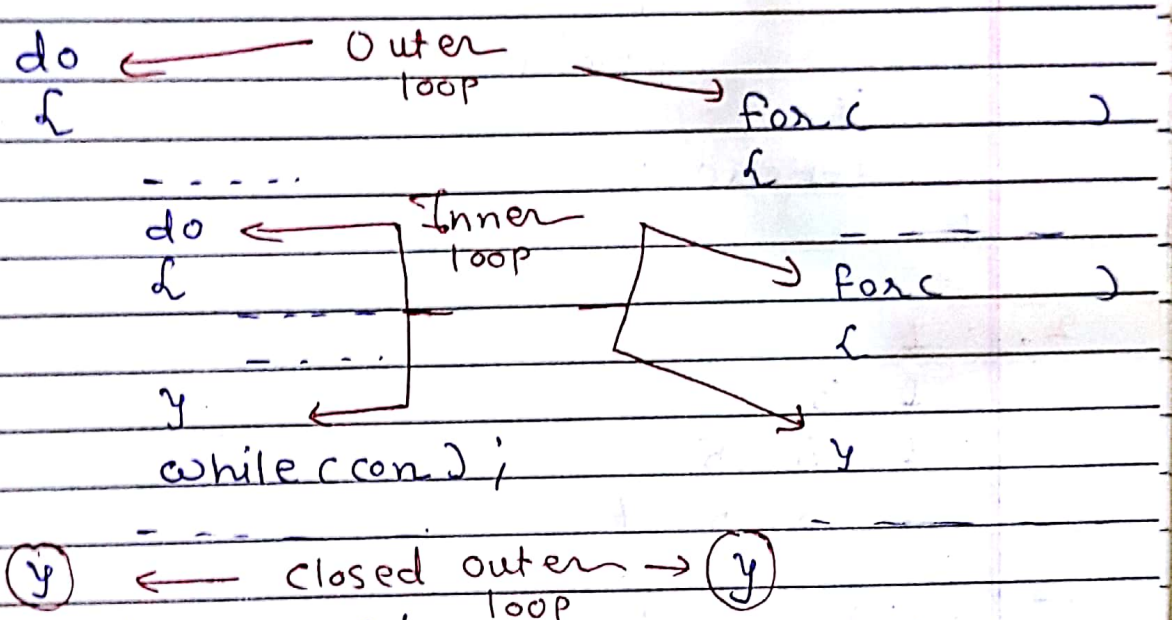
Enter num = 155
Sum of first & last digit = 6

Nested loops:-

↦ A nested loop refers to a loop that is contained within another loop.

↦ In nested loops, the inside loop executes completely before the outside loop's next iteration.

↦ like:-



* WAP to Print following pattern using loop statement for n rows.

I.

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

↦

```
Void main()
{
```

```
int row, col, n;
pfc"Enter num of rows:");
sfc"%d", &n);

for(row=1; row<=n; row++)
{
    for(col=1; col<=row; col++)
    {
        pfc"*");
    }
    pfc"\n");
}
getch();
}
```

2.
```
I
I 3
I 3 5
I 3 5 7
```

↦
```
void main()
{   int row, col, n, m;
    pfc"Enter rows:");
    sfc"%d", &n);

    for(row=1; row<=n; row++)
    {
        m=1;
        for(col=1; col<=row; col++)
        {
            pfc"%d", m);
            m=m+2;
        }
```

```
        pfc("\n");
    }
getch();
}
```

3.
```
1
0 1
1 0 1
0 1 0 1
```

```
void main()
{
    int row, col, n;
    pfc"Enter num of rows:");
    sfc"%d",&n);

    for(row=1;row<=n;row++)
    {
        for(col=1;col<=row;col++)
        {
if((((row%2==0)&&(col%2==0))||((row%2!=0&&
                                col%2!=0)))
  pfc"1");
 else
  pfc"0");
        }
        pfc"\n");
    }
getch();
}
```

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

```
void main()
{
    int row, col, n, k=1;
    pfc"Enter nums of rows:");
    sfc"%d", &n);

    for (row=1; row<=n; row++)
    {
        for (col=1; col<=row; col++)
        {
            pfc"%d", k);
            k++;
        }
        pfc"\n");
    }
    getch();
}
```

5.
```
*
# #
* * *
# # # #
* * * * *
# # # # # #
* * * * * * *
```

```
Void main()
{
    int row, col, n;
    pfc"Enter nums of rows:");
    sfc"%d", &n);

    for(row=1; row<=n; row++)
    {
        for(col=1; col<=row; col++)
        {
            if(row%2!=0)
            pf c"*");
            else
            pf c"#");
        }
        pfc"\n");
    }
  getch();
}
```

6.

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

```
Void main()
{
    int row, col, n;
    pf c"Enter num of rows:");
    sf c"%d", &n);
```

```
for( row = 1; row <= n; row++ )
{
    for( col = 1; col <= row; col++ )
    {
        pf( "%d", row );
    }
    pf( "\n" );
}
getch();
}
```

7.
```
* * * * *
* * * *
* * * *
* *
*
```

```
void main()
{   int row, col, n;
    pf( "Enter num of rows:" );
    sf( "%d", &n );

    for( row = n; row >= 1; row-- )
    {
        for( col = 1; col <= row; col++ )
        {
            pf( "*" );
        }
        pf( "\n" );
    }
getch();
}
```

8. 1 2 3 4 5
2 3 4 5
3 4 5
4 5
5

```
Void main()
{
    int row, col, n, space;
    printf("Enter n:");
    scanf("%d", &n);

    for(row = 1; row <= n; row++)
    {
        for(space = 1; space <= row; space++)
        {
            printf(" ");
        }

        for(col = row; col <= n; col++)
        {
            printf("%d", col);
        }
        printf("\n");
    }
    getch();
}
```

9. 5 4 3 2 1
4 3 2 1
3 2 1
2 1
1

↦
```
Void main()
{
    int row, col, n, space;
    pfc"Enter num of rows:");
    sfc"%d", &n);

  for(row=n; row>=1; row--)
  {
      for(space=1; space<=(n+1-row); space++)
      {
          pfc" ");
      }

      for(col=row; col>=1; col--)
      {
          pfc"%d", col);
      }
      pfc"\n");
  }
getch();
}
```

10.

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

↦
```
Void main()
{
    int row, col, n, space;
    pfc"Enter num of rows:");
    sfc"%d", &n);
```

```
for (row = 1; row <= n; row++)
{
    for(space = 1; space <= (n-row); space++)
    {
        pf("  ");
    }
    for(col = 1; col <= row; col++)
    {
        pf(" * ");
    }
    pf("\n");
}
getch();
}
```

11.

```
        1
    A   B
  1   2   3
A   B   C   D
1 2   3   4   5
```

```
void main()
{
    int row, col, space, n, digit, alpha;
    pf("Enter num of rows:");
    sf("%d", &n);

for(row = 1; row <= n; row++)
{   digit = 1, alpha = 65;
    for(space = 1; space <= (n-row); space++)
    {
        pf("  ");
    }
```

```
for (col = 1 ; col <= row ; col++)
{
    if (row % 2 != 0)
    {
      printf("%d ", digit);
      digit++;
    }
    else
    {
      printf("%c ", alpha);
      alpha++;
    }
  }
  printf("\n");
}
getch();
}
```

12.

```
    1
  A   B
2   3   4
  C   D   E   F
5   6   7   8   9
```

← Try to yourself.

13.

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

Special Control Statements :-

Break Statement	Continue statement
⊢ The break statement is used in loop constructs such as for, while, do & switch statement to terminate execution of the loop or switch.	⊢ The continue statement doesn't terminate the loop but goes to the test exp in the while & do-while statement & then goes to the updating exp. in a for loop.
⊢ It helps to make an early exist from the block where it appears	⊢ It helps in avoiding the remaining statements in a current iteration of the loop & continues with next iteration.
⊢ It can be used in all control statements including switch.	⊢ It can be used only in loop.

⊢ while (test-exp)
{
_ _ _ _ _
_ _ _ _ _
if ( )
break;
}
→ Stop the execution & come at next Statement.
	⊢ while (test_exp)
	{
	_ _ _ _ _
	_ _ _ _ _
	if ( )
	continue;
	} Skipped these part for current iterations.

↦ Example:-

```
void main()
{
    int c=0;
    while (c<=5)
    {
        c++;
        if (c==3)
        ----break;
        pf("\t %d",c);
    }
    getch();
}
```

↦ O/P:- 1 2

↦ Example:-

```
Void main()
{
    int c=0;
↦while (c<=5)
    {
        c++;
        if(c==3)
        ----continue;
        pfc"%d\t",c);
    }
    getch();
}
```

O/P:- 1 2 4 5 6

Prepared By:-
Ms. Jinal Zala
M.E (CE), LJIET