DAY-2

```
#  int main ()                      ~~// j = 0+0~~
   {                                // i 1
    int i = 0, j;
   clrscr ();
     j = i++ + i;
   printf ("%d \t %d", i, j);
    getch ();
    return 0;
   }
```

O/P   1   0

```
#  int main ()                ++i  ++i   --i  i--   --j  --j
                              i - 3 -> 4 -> 3 -> 2   j - 2 -> 1
   {
   int i = 2, j = 3, k;       i 3 -> 4 -> 3, j = 1
    clrsca ()
    k = ++i + ++i + --i - i-- - --j + --j
    k = 3 + 3 + 3 - 3 - 1 + 1
      = 6
   printf (" %d \t %d \t %d , i, j, k);
   getch ();
   return 0;
    }
```

O/P : 2 , 1 , 6.

Date
Page No.

```
# int main ( )
  {
   int i = 1 ;
  clrscr ( ) ;
  if (i++)
  {
   printf ("%d", i ) ;
   }
   getch ( ) ;
   return 0 ;
   }.
```

O/P   2

```
# int ~~[illegible]~~ main ( )
      {
      int x = 1 , y = 2 , z = 3 ;
       switch (x)
       {
         case x :
              printf ("Hello")
         case y :
            printf (" Hii")
         case z
            printf ("Bey") ;
        }
     getch ( ) ;
     return 0 ;
  }.
```

case ke aage hmesha const

Switch case m logical or relational operator use nii ho ~~sk~~ skta

Date
Page No.

~~#~~ Pointer

&, "address of" operator
*, "indirection" operator.

& → variable name ke age likhne se memory location btata

```
# int main ()
  {
  getch ()
   {
   ~~clrscr~~,     int x ;
   ~~get~~          clrscr ();
                 printf ("enter the value of x \n");
                 scanf ("%d", &x );
printf ("%u", &x) ←  getch( );
~~u stand unsigned~~   return 0;
                 }
```

&x → age use hta de value store nii hogi

```
# int main ()
  {
    int x = 5
  clrscr ();
  printf ("%d \t %d", x , *&x ) ;
  getch ();
  return 0;
   }
```

Date / /
Page No.

X=5 $\underbrace{\&x}_{\text{memory location}} = 65324$ $\underbrace{*\&x}_{\text{memory location value kya hai}} = 5$

or

inshort value of

~~indirection → if you put ~~ ~~me before address~~

☆ Pointer → speciable variable: it holds address of a variable

& variable name = address of a variable

datatype variablename ; int x ; X = 5 ;
datatype variable name = value ; int x = 5 ;

datatype * pointername;
datatype * pointername = address of variable;

int * iptr;
float * f ptr;
double * dptr
char * cptr

} declaration .

// (int * iptr = &x)

Date / /
Page No.

```
# int main ()
{
int x = 5
clrscr ()
int *iptr = &x
clrscr ()
printf ("%d t %d\t %d", x, *&x,
                         * iptr);
getch ();
}
O/P  5 5 5
```

&x = iptr.

x = 5
*&x = 5
*iptr = 5
address of x.

```
# int main ()
{
int x = 5
int *iptr = &x;
clrscr()
printf ("%d %d %d %d %d", x, *&x,
             * iptr, **&iptr, *&*iptr);
getch ().
return 0;
}
O/P 5 5 5 5 5
```

```
// ** & iptr
* * iptr = & x.
```

Pointer to pointer Or Double pointer

```
int **dptr,
```

```
# int main
{
 int x = 6;
 int *ptr
iptr = & x;
int **dptr = & iptr;
clrscr ();
printf ("%d", **dptr);
getch ();
 return 0;
}
O/P 6.
```

```
# int main
{
 int x = 6, * iptr = & x;
char ch = 'a', * cptr = & ch;
float f = 5.5, *f ptr = & f;
printf ("%d %d %d", size of (iptr), size of
        (cptr), size of (f ptr));
                 → next (2 files)
getch ();
 return 0;
}
```

O/P 5 5 5

Kise b pointer ki address

```
printf ("%u\t %u", iptr, cptr);
iptr++;
cptr++;
printf("\n %u \t %u", iptr, cptr);
```

```
[illegible]
printf [illegible]
printf [illegible]
  return [illegible]
}
O/P   5
```

int main

```
{
 int i=1;
 clrscr();
 for (++i;++i<10;++i);  {  }
  {
   printf ("%d",++i);
  }
printf ("\n  %d", i);
 getch();
return 0 ;
  }
O/P      12   12
```

★ While Loop

```
initialization
while (cond n);
 {
   ++/--
 }
```

int main

```
 {
    int x =1;              02
    while (++x<=10)
      {
        printf ("%d",x);
    }
 getch();
 return 0;        O/P    2 & 10
}                          O/P  11
```

Date
Page No.

#
```
int main ()
 {
  int count = 1;
   clrscr ();
      while (-- count +1);
    printf ("%d", count);
      getch ();
    return 0;
 }
 O/P = -1.
```

# functions

- Fun declaration
  (main before main ()   void teekam();

- fun call
  inside main   teekam();

- fun def^n
  after main   void teekam () { }

Date
Page No.

# void main teekam(); } If we do not put this then ↑ this define upr

```
int main()
 {
  clrscr ()
  teekam();
  printf ("Thank you \n");
  teekam();
   printf ("all right);
  teekam();
   getch();
 }
void teekam ()
  {
  print f (" jecrc \n");
 }
```

O/P

```
jecrc
Thank You
 ~~all right~~ IE jecrc
 { all right jecrc
```

☆ Based on return type & argument 4 types
return value & no arg
no return type & no arg
return type & args
no return value & arg

0	0
0	1
1	0
1	1

Date
Page No.

```
# void Sum (int x , int y)
  {
      Printf ("%d", x+y);

      }

  int main ()
    {
    int a = 5 , b = 6
    Clrscr(),
    Sum (a,b), ⟶ scanf ("%d %d, &a, &b),
    getch ();
    }

  O/P = 11
        12
        13
        25.
```