POINTERS

Variable Containing add of another Variable.

Pointer is a memory variable that stores a memory address. It is denotd by * operator.

Features:

1. It saves memory space.
2. Execution time with pointer is faster. Direct access (ie. through its address).
3. Declaration

```
int   *x;
float *y;
char  *z;
```

* asterisk symbol.

*(indirection / dereference operator)

```
int a;
```
↳ direct access to val.

```
int *i;
```
indirect access to the value whose address it stores.

* 2 ways of using indirection operator.

(i) declaration ← indicats a pointer variable

(ii) The value stored in the memory location is to be accessed rather than address. during referencing

* Address of variable → To print we use %u.

```
int a=10, *p; int q;
p = &a;   q = *p;
```

a	p	q
10	4000	10
4000	4002	4004

using *p we can access the value 10 (a's value).

Example 1 :

```
Void main ()
{
    int  a = 10, b = 20, *c, *d;  int  add;
    clrscr();
    c = &a;
    d = &b;
    add = a + b;
    printf (" Addition using Variables = %d", add);
    add = *c + *d;
    printf (" Addition using pointers = %d", add);
    getch();
}
```

Output : Addition using Variables = 30
Addition using pointers = 30

Example 2 : Assigning pointer value to another variable

Pointer Arithmetic

Arithmetic operations on pointers is possible :

* Increase
* Decrease
* prefix
* postfix.

Init. Add	Datatype	operation		Add. after operation	
4000	int	++	--	4002	3998
4000	char	++	--	4001	3999
4000	float	++	--	4004	3996
4000	long	++	--	4004	3996

5) Example: Increment & Decrement operators using prefix & postfix

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a=10, *b;
    clrscr();
    b = &a;
    printf("The address of a : %u", b);
    printf("Pre increment a : %u", ++*b);
    printf("post increment a : %u", b++);
    printf("pre decrement a : %u", --b);
    printf("post decrement a : %u", b--);
    getch();
}
```

++a pre increment
a++ post increment
--a pre decrement
a-- post decrement

← b = 4074

← b = 4072

Output:

The address of a = 4072
pre increment a = 4074
post increment a = 4074
pre decrement a = 4074
post decrement a = 4074

Example Programs.

```
Void main()
{
    int  x[5] = {1,2,3,4,5}, k=0;
    clrscr();
    printf ("\n  Index.    Element    Address ");
    while (k <=5)
    {
      printf ("\n  %d  It  %d It  %u ", k, *(x+k), x+k);
      k++;
    }
    getch();
}
```

Output:

Index	Element	Address
0	1	4056
1	2	4058
2	3	4060
3	4	4062
4	5	4064.

Pointer Arithmetic - Example program.

```
# include <stdio.h>
# include <conio.h>
void main()
{  int a=10, b=20, *c, *d;
   c = &a;  d = &b;
   printf ("\n The Add A=%u  B=%u",
                    c, d);
   printf ("\n The Addition = %u", c+1);
   Printf ("\n The Addition = %u", c+2);
   getch();
}
```

A = 65524 ← C

B = 65522 ← D

a	b	c	d
10	20	65524	65522
65524	65522		

c = 65524

c+1 = 65526.

c+2 = 65528

/* the same for subtraction */.

* The multiplication & Division of pointers with constant & with another pointer variable is not possible.
* Addition & Subtraction with constant is possible.
* Subtraction with another pointer variable is possible.

Ex: Addition c + 2 Subtraction c - 2

Subtraction (ptr) c - d ⇒ c = 65524 d = 65522

resultant is 1

* Subtraction Example with array.

a[5] = {1, 2, 3, 4, 5} , *c, *d

c = &a[0]; d = &a[2]

d - c ⇒ Ans is : 2.

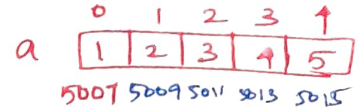
c = 65520
d = 65524

a[0]	a[1]	a[2]
65520	65522	65524

[65524 - 65520 = 4]

Pointer & Arrays - Example.

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a[5] = {1,2,3,4,5},i,*p; int sum=0;
    clrscr();
    p = &a[0]; //p=a;
    for (i=0; i<5; i++)
    {
    printf("\n The Address is %u", (p + i));
    printf("\n The Value is %d", *(p+i));
    sum = sum + *(p+i);
    }
printf("\n The Sum = %d", sum);
getch();
}
```



*p. *(p+1)

&a[0] a

a = 5007

```
i=0, sum=0;
while (i<5)
{
    sum = sum + *p;
    i++; p++;
}
printf("%d", sum);
```