**Week 2 assignment 1**

**Explain pointers and its operation (& and \*) with proper explanation.**

A pointer is a variable which contains the address in memory of another variable or function.

& - address of a variable

\* - contents of an object pointed to by a pointer

**Explain pointer arithmetic. (++ and --).**

++ -> If we increment a pointer by 1, the pointer will start pointing to the immediate next location. This is somewhat different from the general arithmetic since the value of the pointer will get increased by the size of the data type to which the pointer is pointing.

--> Like increment, we can decrement a pointer variable. If we decrement a pointer, it will start pointing to the previous location.

**Explain array name as pointers.**

The array name can be used to access all the elements of the array. The only different between an array name and a normal pointer variable is that an array name always points to a specific address, which is the starting address of the array. Pointers are flexible and can point to where ever you want them to point.

OUTPUT

1. Ans:- 20

Because here fun(y) call the value of x, the value of y is 20 . In printf function we are printing the value of y. so, 20 will print in output.

1. Ans:- 30

Here &y functions calls the value of \*ptr which is sore in y so y value get updated, we are printing value of y i.e 30

1. Ans- x =0 //ptr =&x , \*ptr=o, from these two \*&x=0 which is indicating value of x=0

\*ptr = 0 // it is defined so, value is 0

x =5

\*ptr = 5 // increment in x is 5 so value of x is change to 5 and also \*ptr increases by 5

x =6

\*ptr = 6 // increment in x is 1 so value of x is change to 6 and also \*ptr increases by 1

1. Ans:- 1204

char s1[7] = "1234", \*p;

p = s1 + 2; // p holds address of character 3

\*p = '0' ; // memory at s1 + 3 now becomes 0

printf ("%s", s1); // All characters are printed

1. Ans:- 0 , 2

I.e value of i is 0 and j is 2

1. Ans- 19

Here x+y+z=19

1. Ans:- 3

It will print 3 because \*p and arr have same index value and there is an increment of 2 so on the third index value 3 is there so 3 will print.

1. Ans:- 2011

char c[] = "GATE2011";

// p now has the base address string "GATE2011"

char \*p =c;

// p[3] is 'E' and p[1] is 'A'.

// p[3] - p[1] = ASCII value of 'E' - ASCII value of 'A' = 4

// So the expression p + p[3] - p[1] becomes p + 4 which is

// base address of string "2011"

1. #include <stdio.h>

int main()

{

char arr[]="workstreet";

printf("%s", arr + arr[4] - arr[1]);

return 0;

}

1. Ans 10