Pointers

A Pointer in C language is a variable which holds the address of another variable of the same data type.

Pointers are used to access memory and manipulate the address.

Pointers are one of the most distinct and exciting features of the C language. It provides power and flexibility to the language.

Address in C

Whenever a variable is defined in C language, a memory location is assigned for it, in which its value will be stored. We can easily check this memory address, using the & symbol.

If var is the name of the variable, then &var will give its address.

Let's write a small program to see the memory address of any variable that we define in our program.

```
#include<stdio.h>

int main()

{

   int var = 7;

   printf("Value of the variable var is: %d\n", var);

   printf("Memory address of the variable var is: %x\n", &var);

   return 0;

}
```

Output:

```
Value of the variable var is: 7

Memory address of the variable var is: bcc7a00
```

You must have also seen in the function scanf(), we mention &var to take user input for any variable var.

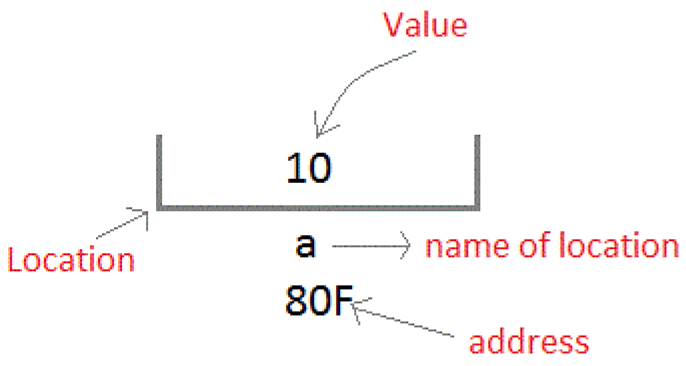
```
scanf("%d", &var);
```

This is used to store the user inputted value to the address of the variable var.

Concept of Pointers

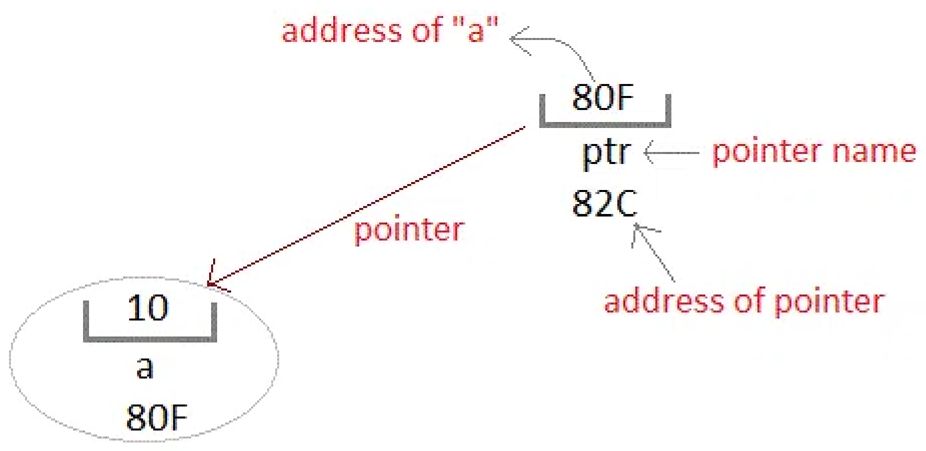
Let us assume that the system has allocated memory location 80F for a variable a.

int a = 10;



We can access the value 10 either by using the variable name a or by using its address 80F.

A pointer variable is therefore nothing but a variable which holds an address of some other variable. And the value of a pointer variable gets stored in another memory location.



Benefits of using pointers

Below we have listed a few benefits of using pointers:

1. Pointers are more efficient in handling arrays and structures.
2. Pointers allow references to functions and thereby help in passing functions as arguments to other functions.
3. It reduces the length of the program and its execution time as well.
4. It allows the C language to support Dynamic Memory management.

Declaring, Initializing and using a pointer variable in C

Declaration of C Pointer variable

The general syntax of pointer declaration is,

```
datatype *pointer_name;
```

The data type of the pointer and the variable to which the pointer variable is pointing must be the same.

Pointer Syntax

Here is how we can declare pointers.

```
int* p;
```

Here, we have declared a pointer p of int type.

You can also declare pointers in these ways.

```
int *p1;

int * p2;
```

Let's take another example of declaring pointers.

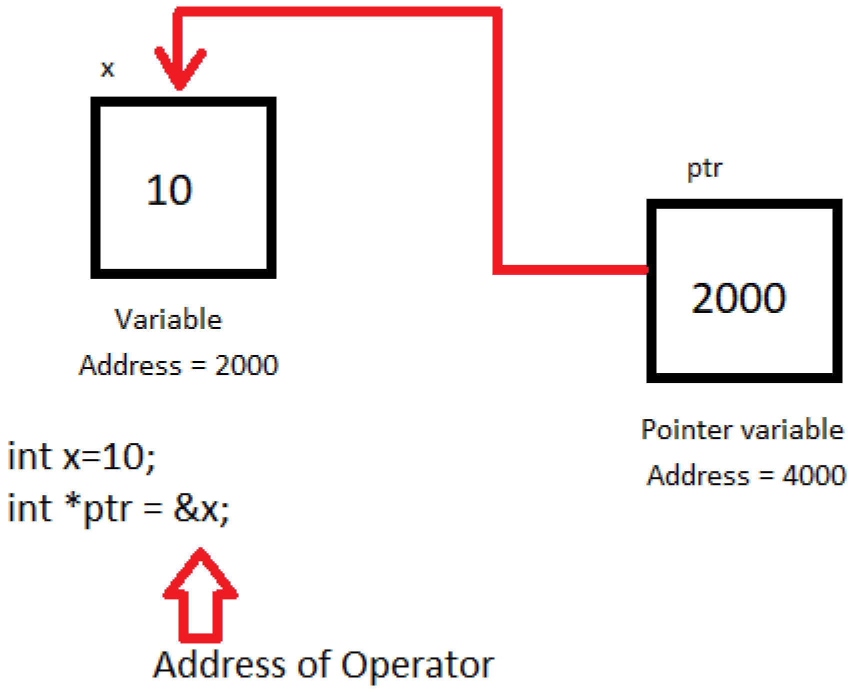
```
int* p1, p2;
```

Here, we have declared a pointer p1 and a normal variable p2

Initialization of C Pointer variable

Pointer Initialization is the process of assigning the address of a variable to a pointer variable. It contains the address of a variable of the same data type. In C language address operator & is used to determine the address of a variable. The & (immediately preceding a variable name) returns the address of the variable associated with it.

```
int a = 10;

int *ptr;       //pointer declaration

ptr = &a;       //pointer initialization
```



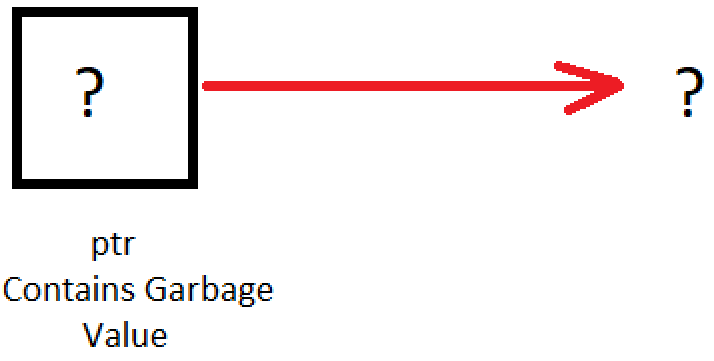
```
int x=10;
int *ptr = &x;
```

Pointer variable always points to variables of the same datatype. For example:

```
float a;
```

```
int *ptr = &a;       // ERROR, type mismatch
```

While declaring a pointer variable, if it is not assigned to anything then it contains garbage value. Therefore, it is recommended to assign a NULL value to it,



A pointer that is assigned a NULL value is called a Null pointer in C.

```
int *ptr = NULL;
```

Once a pointer has been assigned the address of a variable, to access the value of the variable, the pointer is dereferenced, using the **indirection operator or dereferencing operator** *

Points to remember while using pointers

- While declaring/initializing the pointer variable, * indicates that the variable is a pointer.
- The address of any variable is given by preceding the variable name with Ampersand &.

- The pointer variable stores the address of a variable.
- The declaration int *a doesn't mean that a is going to contain an integer value. It means that a is going to contain the address of a variable storing integer value.
- To access the value of a certain address stored by a pointer variable * is used. Here, the * can be read as 'value at'.

1.Write a program to show the basic declaration of a pointer and use of & and * operator.

```
#include <stdio.h>

int main()

{

   int a;

   a = 10;

   int *p = &a;     // declaring and initializing the pointer

   //prints the value of 'a'

   printf("%d\n", *p);

   printf("%d\n", *&a);

   //prints the address of 'a'

   printf("%u\n", &a);

   printf("%u\n", p);

   printf("%u\n", &p);    //prints address of 'p'

   return 0;
```

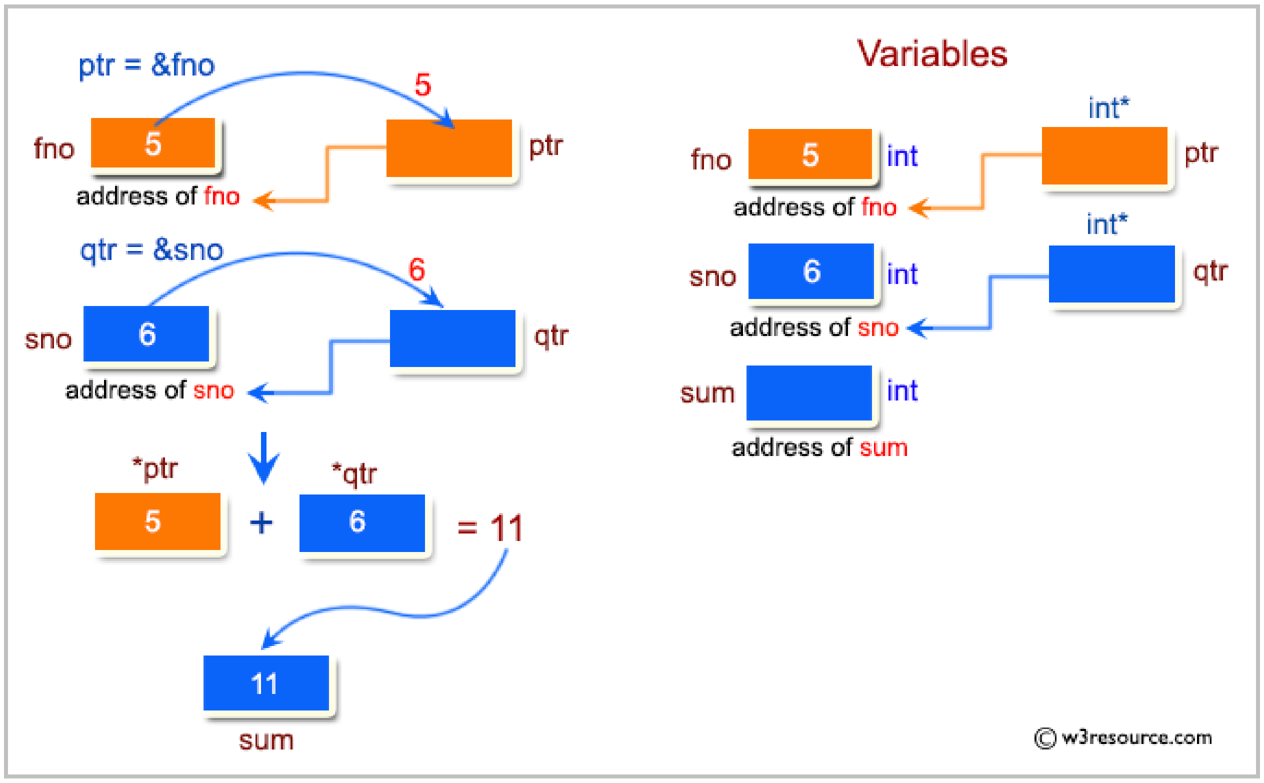
}

Output:

```
10

10

6684188

6684188

6684176
```

2.Write a program to find the sum of 2 numbers using a pointer.

Pictorial Presentation:



#include <stdio.h>

```
int main()
{
   int fno, sno, *ptr, *qtr, sum;
   printf("\n\n Pointer : Add two numbers :\n");
   printf("--------------------------------\n");
   printf(" Input the first number : ");
   scanf("%d", &fno);
   printf(" Input the second  number : ");
   scanf("%d", &sno);
   ptr = &fno;
   qtr = &sno;
   sum = *ptr + *qtr;
   printf(" The sum of the entered numbers is : %d\n\n",sum);
   return 0;
}
```

Output:

```
 Pointer : Add two numbers :
--------------------------------
 Input the first number : 5
 Input the second  number : 6
 The sum of the entered numbers is : 11
```

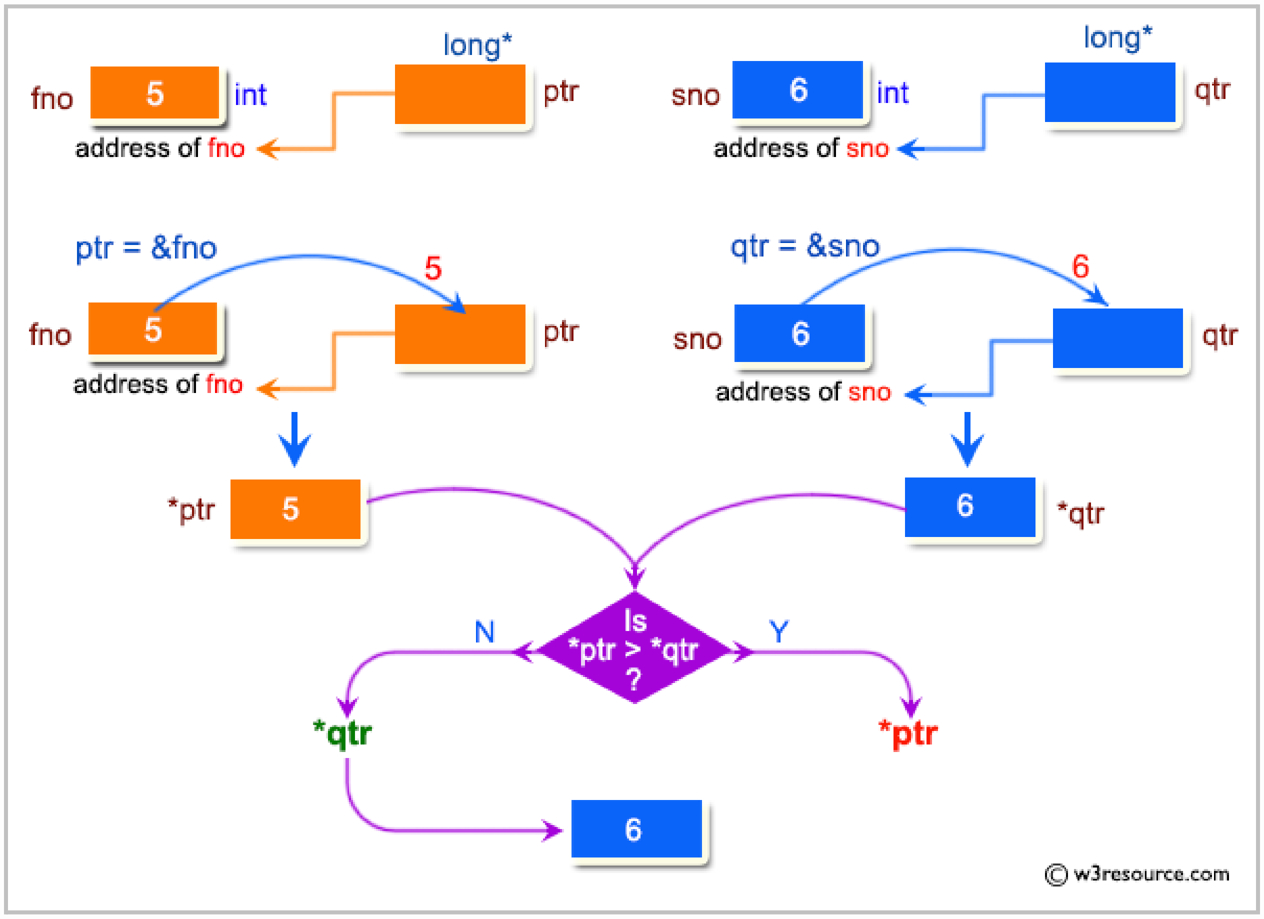
3.Write a program to perform all the arithmetic operations using pointer

```
#include <stdio.h>
int main()
{
  int fno, sno, *ptr, *qtr, sum,subtract,mul,div;
  printf("\n\n Pointer : Add two numbers :\n");
  printf("--------------------------------\n");
  printf(" Input the first number : ");
  scanf("%d", &fno);
  printf(" Input the second  number : ");
  scanf("%d", &sno);
  ptr = &fno;
  qtr = &sno;
  sum = *ptr + *qtr;
  subtract=*ptr - *qtr;
  mul=*ptr * *qtr;
  div=*ptr / *qtr;

       printf(" The sum of the entered numbers is : %d\n\n",sum);

     printf(" The subtraction of the entered numbers is : %d\n\n",subtract);

    printf(" The multiplication of the entered numbers is : %d\n\n",mul);

      printf(" The division of the entered numbers is : %d\n\n",div);

  return 0;

}
```

Output:

```
 Pointer : Add two numbers :

--------------------------------

 Input the first number : 6

 Input the second  number : 3

 The sum of the entered numbers is : 9

 The subtraction of the entered numbers is : 3

 The multiplication of the entered numbers is : 18

 The division of the entered numbers is : 2
```

4.Write a program to find the greatest among 2 numbers using a pointer.



```
#include <stdio.h>

int main()

{

 int fno,sno,*ptr1=&fno,*ptr2=&sno;

   printf("\n\n Pointer : Find the maximum number between two numbers :\n");


printf("--------------------------------------------------------\n");
```

```
  printf(" Input the first number : ");
  scanf("%d", ptr1);
  printf(" Input the second  number : ");
  scanf("%d", ptr2);
 if(*ptr1>*ptr2)
 {
  printf("\n\n %d is the maximum number.\n\n",*ptr1);
 }
 else
 {
  printf("\n\n %d is the maximum number.\n\n",*ptr2);
 }
 return 0;
}
```

Output:

```
 Pointer : Find the maximum number between two numbers
:
------------------------------------------------------------
 Input the first number : 5
 Input the second  number : 6
```

6 is the maximum number.