```
void fun(char a[])  // here a is a constant pointer which now contains the address
b
{   strcpy(a,"xyz");  // contents of  address stored in a is changed
    cout<<"\n in function a="<<a;
}// during 1st call contents of 1000 is replaced and 2nd call contents of address
2001.
void main()
{   clrscr();
    char b[20]="abc";  // Assuming address of abc as 1000
 // here an array is created and abc is stored and it base address is stored in a
constant pointer b
    fun(b);                                                    // Address stored in b is
passed . ie 1000
    cout<<"\n in main after calling b="<<b;
    char c[20]="abc"; // here c is a constant pointer
    fun(c+1); // if c contains address 2000 then 2001 is passed to function
    cout<<"\n in main after calling c="<<c;
}
```

Output

in function a=xyz
in main after calling b=xyz
in function a=xyz
in main after calling c=axyz

```
void fun(char *a) // char a[]
{    strcpy(a,"xyz");
     cout<<"\n in function a="<<a;
}
void main()
{    clrscr();
     char *b="abcdefgh"; // here b is a pointer variable
     fun(b);
     cout<<"\n in main after calling b="<<b;
}
```

Output:

in function a=xyz
in main after calling b=xyz

```
void fun(char *a) // char a[] is an error, here a is pointer variable and a contains
address of string abc
{  a="uvw"; // now address of string uvw is stored in a
   cout<<"\n in function a="<<a;
}
void main()
{  clrscr();
   char *b="abc";  // now address of string abc is stored in b
    fun(b); // b is passed to a
```

```
    cout<<"\n in main after calling b="<<b;
}
```
Output
in function a=uvw
in main after calling b=abc
```
void fun(char **a)
{ char *b="pqrst";
  *a=b;
  cout<<"\n in function a="<<*a<<" b="<<b;
}
void main()
 {char *c="abcdef";
  fun(&c); // here address of c is passed
  cout<<"\n in main after calling c="<<c;
}
```
Output:
in function a=pqrst b=pqrst
in main after calling c=pqrst

```
void main()
{char *p="aeiou";
```
char q=++*p++; // same output for q=++(*p++),
```
// contents of p is incremented and stored in q and then p is incremented
cout<<"\n"<<q;//b
cout<<"\n"<<p;//eiou
cout<<"\n"<<*p;//e
}
```
Output
b
eiou
e
```
void main()
{char *p="aeiou";
```
char q=(++*p)++; // no change in address stored in p
```
// contents of p is incremented twice
cout<<"\n"<<q;   //b
cout<<"\n"<<p;   //ceiou
cout<<"\n"<<*p;  //c
}
```
Output
b
ceiou
c
Note: the operation ++(*p)++ is a syntax error
```
void main()
{          int b[]={11,22,33,44,55};
        int *q=b;
```

```
        int y=*q++;  //y=*q, q=q+1
        cout<<*q<<"  "t<<y;
}
```
output
22   11
```
void main()
{               int b[]={11,22,33,44,55};
           int *q=b;
           int y=(*q)++;  //y=*q,*q=*q+1;
           cout<<*q<<"     "<<y;
}
```
output
12   11
```
void main()
{                int b[]={11,22,33,44,55};
            int *q=b;
            int y=*++q;  //q=q+1,y=*q;
            cout<<*q<<"   "t<<y;
}
```
output
22   22
```
void main()
{               int b[]={11,22,33,44,55};
           int *q=b;
           int y=++*q;  //*q=*q+1,y=*q;
           cout<<*q<<"   "<<y;
}
```
output
12   12

Pointers to functions

```
int add(int a,int b)
{return a+b;
}
int sub(int a,int b)
{return a-b;
}
void main()
{
```
int (*p)(int,int); //here p is not a function but a pointer variable
/* The () indicates that it is a pointer to a function p can hold address of any function having return datatype int and having two arguements of type int. it is different from int *p(int,int); which means p is a function having return datatype int* and two arguements of type int. */
```
int x=23,y=5;
```
p=&add;   // or p=add;
cout<<(*p)(x,y)<<"\n";   //or cout<<p(x,y)<<"\n";
```
p=sub;
```

```
cout<<p(x,y);
}
```
Output
28
18