Loop interruption

Loop interruption

It is sometimes convenient to be able to exit from a loop other than by testing the loop termination condition at the top or bottom.

A. Jumping out of a loop

```
    break;
```

B. Skipping a part of a loop

```
    continue;
```

Program to Implement *break Statement*

```
main()
{
int i;
for (i=1; i<=10; i++)
{
cout<<i <<"\n";
if (i == 7)
break;
} }
```

Printing as between 1 to 10

```
void main()
    {
int  i;
for(i=1;i<=10; i++)
            {
           if(i==3)
                     {
                     continue;
                     }
    if(i==8)
             {
              break;
              }
        cout<<"\t"<<i<<endl; }}
```

Program to Implement *continue Statement*

```
void main()
{
int i,n,a,sq;
cout<<"\n square of positive No. only\n";
cout<<"\nHow many times you want to square:";
cin>>n;
for(i=1;i<=n;i++)
{
cout<<"Enter a No.\n";
cin>>a;
if(a<0)
continue;
sq=a*a;
cout<<"\nsquare=\n"<<sq<<endl; } }
```

The exit function

- The standard library function, exit ( ), is used to terminate execution of the program.
- The difference between break statement and exit function is, break just terminates the execution of loop in which it appears,
- whereas exit ( ) terminates the execution of the program itself.

Example - Exit() Function

```
#include<iostream.h>
#include<stdlib.h>
#include<conio.h>
void main()
{
clrscr();
cout<<"start of the program....\n"<<endl;
cout<<"Exiting the program....\n"<<endl;
exit(0);
cout<<"End of the program....\n"<<endl;
getch();
}
```

Thank You!