

005 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.

1. **break;** **Jumping out of a loop.**
2. **continue;** **Skipping a part of a loop.**
3. **exit();** **Exit out From the Whole Program**

Program to Implement *break* Statement

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

continue Statement

```
main()
{
    int i;
    for (i=101; i<=110; i++)
    {
        if (i == 107)
        {
            continue;
        }
        printf("\n%d", i);
    }
}
```

Printing as between 1 to 10

```
Void main()
```

```
{
```

```
int i;
```

```
for(i=1;i<=10; i++)
```

```
{
```

```
if(i==3)
```

```
{
```

```
continue;
```

```
}
```

```
if(i==8)
```

```
{
```

```
break;
```

```
}
```

```
printf("\t%d",i); }}
```

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 <stdio.h>
#include <stdlib.h>
Void main ()
{
printf("Start of the program....\n");
printf("Exiting the program....\n");
exit(0);
printf("End of the program....\n");
}
```

Thank You!

The image features the text "Thank You!" in a bold, white, three-dimensional sans-serif font. The text is centered and appears to be floating above or attached to a dynamic, multi-colored brushstroke. The brushstroke is composed of numerous overlapping, curved strokes in a spectrum of colors including red, orange, yellow, green, blue, and purple, creating a sense of movement and energy. The entire graphic is set against a plain white background.