Tutorial Link https://codequotient.com/tutorials/Iteration - Loop control with break and continue/5a22a852c66cfe38f29622ae

TUTORIAL

Iteration - Loop control with break and continue

Chapter

In loops, we came across the situations where we need to come out of loop instantly instead the usual condition way. We have two keywords break and continue for this purpose. Break will cause the control to go after the loop body, whereas continue will make control to go at beginning of loop condition. Both serve two different purposes. We generally use both with an if statement inside the loop body. For example,

This is a simple for loop to print the values from 1 to 10: -

```c
#include <stdio.h>

int main()
{
  int var1;
```

```
  for(var1 = 1; var1 <= 10; var1++)
  {
    printf("var1 = %d\n", var1);
  }
  printf("After loop var1 = %d \n",var1);
  return 0;
}

```

Output: -

```
var1 = 1
var1 = 2
var1 = 3
var1 = 4
var1 = 5
var1 = 6
var1 = 7
var1 = 8
var1 = 9
var1 = 10
After loop var1 = 11
```

If we use a break statement, when var1 becomes 5, as below: -

break Implementation

```c
#include <stdio.h>

int main()
{
  int var1;
  for(var1 = 1; var1 <= 10; var1++)
  {
    if(var1==5)
      break;
    printf("var1 = %d\n", var1);
  }
```

```
  printf("After loop var1 = %d \n",var1);
  return 0;
}

```

Output: -

```
var1 = 1
var1 = 2
var1 = 3
var1 = 4
After loop var1 = 5
```

instead of break if use continue statement: -

continue Implementation

```c
#include <stdio.h>

int main()
{
  int var1;
  for(var1 = 1; var1 <= 10; var1++)
  {
    if(var1==5)
      continue;
    printf("var1 = %d\n", var1);
  }
  printf("After loop var1 = %d \n",var1);
  return 0;
}

```

Output: -

```
var1 = 1
var1 = 2
```

```
var1 = 3
var1 = 4
var1 = 6
var1 = 7
var1 = 8
var1 = 9
var1 = 10
After loop var1 = 11
```

From above example, it will be clear where the control moves with break and continue statements.

If we use break and continue in nested loops, then they will be associated with innermost loops. So they cause the control out of inner loops only. For example,

break and continue in nested loops

```c
#include <stdio.h>

int main()
{
  int var1, var2;
  for(var1 = 1; var1 <= 5; var1++)
  {
    for(var2 = 1; var2 <= 5; var2++)
    {
      if(var2==3)
        break;  // it will cause exit from inner loop.
      printf("var1 = %d, var2 = %d\n", var1, var2);
    }
    printf("After second loop.\n"); // it will execute after the break statement. as it is in outer loop.
  }
  printf("After both loops.");
  return 0;
}

```