Continue:

When we execute the loop

It will usually execute upto last line and again come back to initial point of the loop like

-> Loop (){

| —

| —

| —

| —

| —

| - }

But when we write continue anywhere in loop body it will take control to the start of the loop.

-> Loop (){

| —

| —

| —

| - continue;

—

—

}

In case of while loop

It will go to top of the loop directly

In case of for loop if continue is written

Then 3rd statement will execute then it will go to start

In case of do while when we write continue

Then 1st it will validate the while condition if true then only it will come back to top otherwise it will termintte the loop if condition is wrong.

Without continue:

#include<stdio.h>

int main(){

int x;

x = 1;

while(x<=15){

if(x%4 != 0){

printf("%d\n", x);

}

x++;

printf("cool\n");

}

return 0;

}

Output:

1

cool

2

cool

3

cool

cool

5

cool

6

cool

7

cool

cool

9

cool

10

cool

11

cool

cool

13

cool

14

cool

15

cool

With Continue:

#include<stdio.h>

int main(){

int x;

x = 1;

while(x<=15){

if(x%4 != 0){

printf("%d\n", x);

x++;

continue;

}

x++;

printf("cool\n");

}

return 0;

}

Output:

1

2

3

cool

5

6

7

cool

9

10

11

cool

13

14

15

`**continue**` Keyword Behavior in Different Loops

The `continue` keyword behaves differently depending on the type of loop in which it is used:

1. `while` Loop: When `continue` is encountered, the loop skips the remaining code inside the loop and goes back to the top to re-evaluate the condition.

2. `for` Loop: When `continue` is encountered, the loop skips the remaining code, executes the third expression (usually used for incrementing or updating the loop counter), and then re-evaluates the condition.

3. `do-while` Loop: When `continue` is encountered, the loop skips the remaining code, checks the `while` condition at the end of the loop, and if the condition is true, it goes back to the top of the loop.

In summary, `continue` causes the loop to skip to the next iteration:

- In a `while` loop, it goes directly to the condition check.

- In a `for` loop, it first executes the update expression, then checks the condition.

- In a `do-while` loop, it checks the condition at the end before continuing.