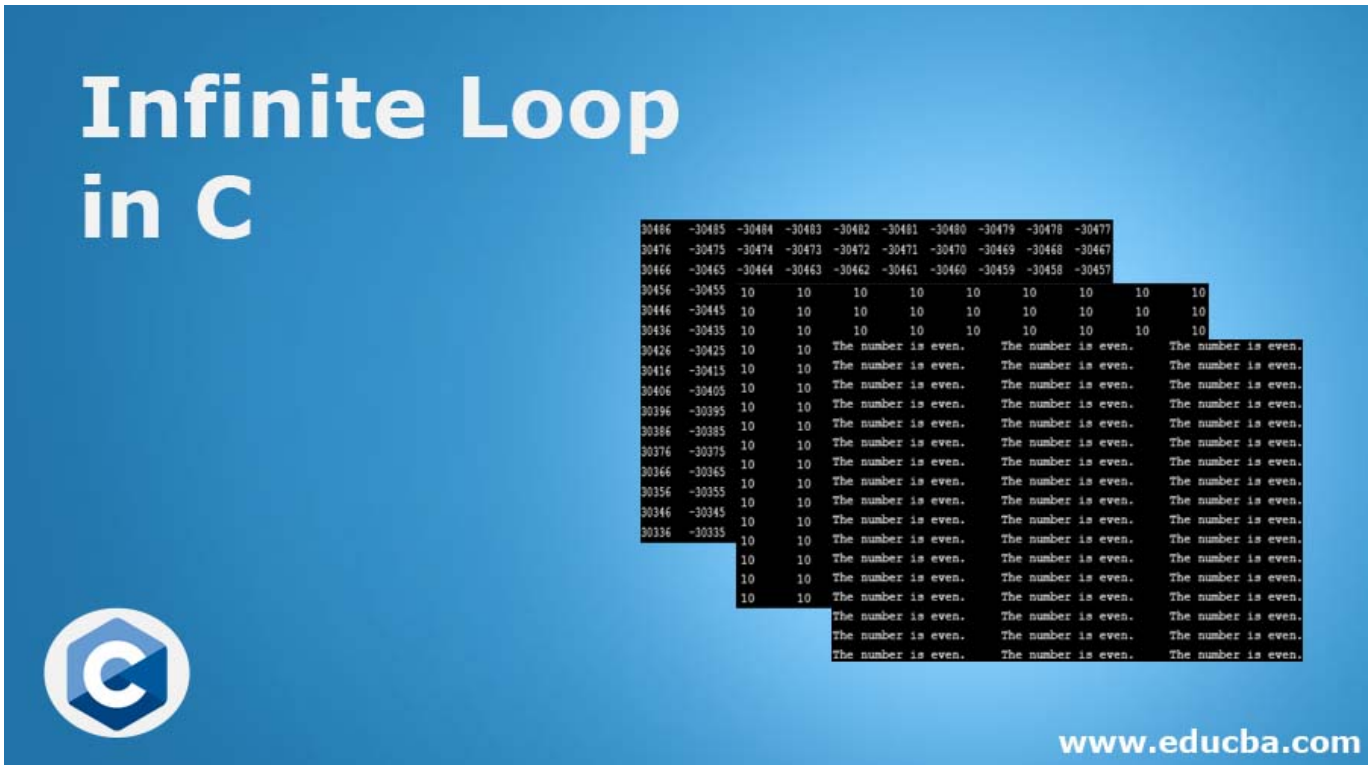
EDUCBA
(https://www.educba.com/software-development/)

← (https://www.educba.com/nested-loop-in-c/)

→ (https://www.educba.com/c-string-functions/)



Introduction to Infinite Loop in C

A loop that repeats indefinitely and does not terminate is called an infinite loop. An infi
also called as endless loop or indefinite loop. An infinite loop is most of the time create by the mistake, but it does not mean that infinite loop is not require or not useful. Infinite loop can be

(https://www.educba.com/software-development/)

The infinite loop in a program can be created in two ways:

1. Unintentionally
2. Intentionally

Unintentionally infinite loop gets create by bug in the code, by mistake or by specifying the condition which never becomes false. And intentionally infinite loop explicitly creates to achieve some requirement in an application. The loop structures we can use to create intentionally or explicitly infinite loop and run the code specified in a loop to repeatedly or infinite times. So we can use the following loops do create an infinite loop –

1. for loop
2. while loop
3. do-while loop
4. go to statement
5. C macros

1. For loop

Syntax:


```
for( ; ; )
{
// some code which run infinite times
```

(https://www.educba.com/software-development/)

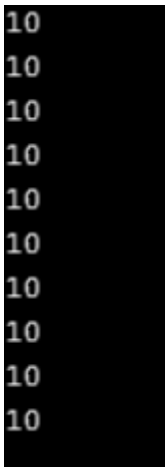
infinite times.

Next, we write the c code to understand the infinite for loop working more clearly with the following example.

Code:

```
#include <stdio.h>
void main()
{  int i = 10;
for( ; ;)
{
printf("%d\n",i);
}
}
```

Output:



As in the above code the for loop is running for infinite times and printing the i value that is 10 infinitely.

Next we write the c code to show the kind of mistakes can lead to an infinite loop in for loop –

Code:

```
#include <stdio.h>
void main()
{  short int x;
```

```
}
```

Output:

```
30486   -30485  -30484  -30483  -30482  -30481  -30480  -30479  -30478  -30477
30476   -30475  -30474  -30473  -30472  -30471  -30470  -30469  -30468  -30467
30466   -30465  -30464  -30463  -30462  -30461  -30460  -30459  -30458  -30457
30456   -30455  -30454  -30453  -30452  -30451  -30450  -30449  -30448  -30447
30446   -30445  -30444  -30443  -30442  -30441  -30440  -30439  -30438  -30437
30436   -30435  -30434  -30433  -30432  -30431  -30430  -30429  -30428  -30427
30426   -30425  -30424  -30423  -30422  -30421  -30420  -30419  -30418  -30417
30416   -30415  -30414  -30413  -30412  -30411  -30410  -30409  -30408  -30407
30406   -30405  -30404  -30403  -30402  -30401  -30400  -30399  -30398  -30397
30396   -30395  -30394  -30393  -30392  -30391  -30390  -30389  -30388  -30387
30386   -30385  -30384  -30383  -30382  -30381  -30380  -30379  -30378  -30377
30376   -30375  -30374  -30373  -30372  -30371  -30370  -30369  -30368  -30367
30366   -30365  -30364  -30363  -30362  -30361  -30360  -30359  -30358  -30357
30356   -30355  -30354  -30353  -30352  -30351  -30350  -30349  -30348  -30347
30346   -30345  -30344  -30343  -30342  -30341  -30340  -30339  -30338  -30337
30336   -30335  -30334  -30333  -30332  -30331  -30330  -30329  -30328  -30327
```

As above the loop is running infinite times because short int ranges is -32768 to 32767, so when i is the increment above to 32767 it becomes negative and hence the condition becomes always true.

2. While Loop

Syntax:


```
while(1)
{
// some code which run infinite times
```

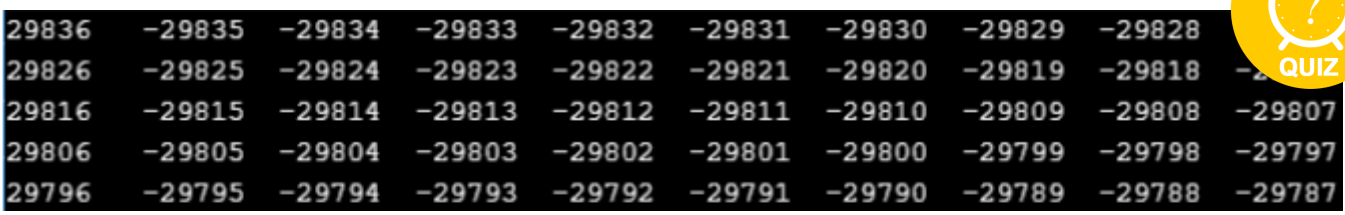
(https://www.educba.com/software-development/)

Next we write the c code to create the infinite loop by using while loop with the following example.

Code:

```
#include <stdio.h>
void main()
{  int i = 10;
while(1)
{
printf("%d\t",i);
i++;
}
}
```

Output:



EDUCBA
(https://www.educba.com/software-development/)

```
29716   -29715  -29714  -29713  -29712  -29711  -29710  -29709  -29708  -29707
29706   -29705  -29704  -29703  -29702  -29701  -29700  -29699  -29698  -29697
29696   -29695  -29694  -29693  -29692  -29691  -29690  -29689  -29688  -29687
29686   -29685  -29684  -29683  -29682  -29681  -29680  -29679  -29678  -29677
29676   -29675  -29674  -29673  -29672  -29671  -29670  -29669  -29668  -29667
29666   -29665  -29664  -29663  -29662  -29661  -29660  -29659  -29658  -29657
29656   -29655  -29654  -29653  -29652  -29651  -29650  -29649  -29648  -29647
29646   -29645  -29644  -29643  -29642  -29641  -29640  -29639  -29638  -29637
```

As in the above code while loop runs infinite times because the condition always becomes true and the i value is updated infinite times.

Next we write the c code to show the kind of mistakes can lead to an infinite loop in for loop –

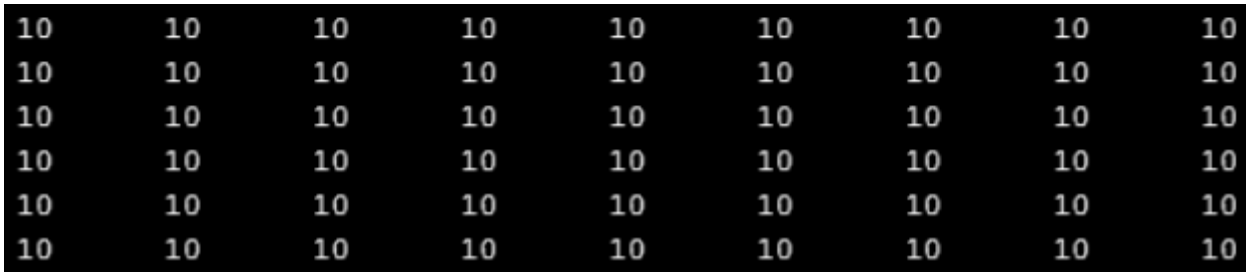
Code:

```
#include <stdio.h>
void main()
{  int i = 10;
while(i<100)
{
printf("%d\t",i);
}
}
```

Output:

```
10      10      10      10      10      10      10      10      10
10      10      10      10      10      10      10      10      10
10      10      10      10      10      10      10      10      10
10      10      10      10      10      10      10      10      10
10      10      10      10      10      10      10      10      10
```

(https://www.educba.com/software-development/)



As in the above code the mistake is updating of I value is missing which leads to an infinite loop.

Other than this some more mistake which can lead to an infinite loop are:

- If Semicolon placed in the wrong position may lead to an infinite loop.

Example:

```
while(cond);
{
//code
}
```

- If logical conditions wrong by mistake, we used assignment operator (=) instead of a relational operator (= =) may lead to an infinite loop.

Example:


```
while(inp='y')
{
//code
```

(https://www.educba.com/software-development/)

Example:

```
for(int i=0;i>=0;i++)
{
//code
}
```

3. Do-While Loop

Syntax:

```
do
{
// some code which run infinite times
}  while(1);
```

Next we write the c code to create the infinite loop by using do-while loop with the following example.

Code:


```
#include <stdio.h>
void main()
{  int i = 10;
```

(https://www.educba.com/software-development/)

```
}  while(i);
}
```

Output:

```
75100   175101  175102  175103  175104  175105  175106  175107  175108  175109
75110   175111  175112  175113  175114  175115  175116  175117  175118  175119
75120   175121  175122  175123  175124  175125  175126  175127  175128  175129
75130   175131  175132  175133  175134  175135  175136  175137  175138  175139
75140   175141  175142  175143  175144  175145  175146  175147  175148  175149
75150   175151  175152  175153  175154  175155  175156  175157  175158  175159
75160   175161  175162  175163  175164  175165  175166  175167  175168  175169
75170   175171  175172  175173  175174  175175  175176  175177  175178  175179
75180   175181  175182  175183  175184  175185  175186  175187  175188  175189
75190   175191  175192  175193  175194  175195  175196  175197  175198  175199
75200   175201  175202  175203  175204  175205  175206  175207  175208  175209
75210   175211  175212  175213  175214  175215  175216  175217  175218  175219
75220   175221  175222  175223  175224  175225  175226  175227  175228  175229
75230   175231  175232  175233  175234  175235  175236  175237  175238  175239
75240   175241  175242  175243  175244  175245  175246  175247  175248  175249
```

4. Goto Statement

Syntax:

```
label:
// some code which run infinite times
goto label;
```

Next we write the c code to create the infinite loop by using goto statement with the fo
example.

Code:

(https://www.educba.com/software-development/)

```
if (num%2 == 0)
goto even_no;
else
goto odd_no;
even_no:
printf("The number is even.\t");
goto even_no;
odd_no:
printf("The number is odd.\t");
goto odd_no;
}
void main() {
int i = 10;
checkEven(i);
}
```

Output:

```
The number is even.     The number is even.     The number is even
The number is even.     The number is even.     The number is even.
The number is even.     The number is even.     The number is even.
The number is even.     The number is even.     The number is even.
The number is even.     The number is even.     The number is even.
```

(https://www.educba.com/software-development/)

```
The number is even.     The number is even.     The number is even.
The number is even.     The number is even.     The number is even.
The number is even.     The number is even.     The number is even.
The number is even.     The number is even.     The number is even.
The number is even.     The number is even.     The number is even.
The number is even.     The number is even.     The number is even.
```

As in the above code the goto statement becomes the infinite loop.

5. Macros

To create the infinite loop we can use macro which defines the infinite loop. Next we write the c code to create the infinite loop by using macro with the following example.

Code:

```
#include<stdio.h>
#define macro_name for( ; ; )
void main()
{
int i=10;
macro_name
{
printf("%d\t", i);
}
}
```

Output:

```
10      10      10      10      10      10      10      10      10
10      10      10      10      10      10      10      10      10
```

(https://www.educba.com/software-development/)

```
10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10
```

As in the above code the macro is defined whose value is for(;;). Later in a main function macro is used by its name, whenever the name of macro comes it gets replaced by its value.

Conclusion

An infinite loop is a loop that repeats indefinitely and does not terminate. A program can have infinite loop by intentionally or unintentionally as we have seen above. We have seen various ways to create an infinite loop and the solution to come out from infinite loop is use of break statement.

Recommended Articles

This is a guide to Infinite Loop in C. Here we discuss the Introduction to Nested Loop in C and its working along with the examples and code implementation. You can also go through our other suggested articles to learn more –

1. Prime Numbers in C (Examples) (https://www.educba.com/prime-numbers-in-c/)
2. How to Reverse Number in C? (https://www.educba.com/reverse-number-in-c/)
3. Introduction to Reverse String in C (https://www.educba.com/reverse-string-in-c/)
4. Prime Numbers in Java | Top 3 Examples (https://www.educba.com/prime-numbers-in-java/)

(https://www.educba.com/software-development/)

source=footer)

Privacy Policy (https://www.educba.com/privacy-policy/?source=footer)

Apps

iPhone & iPad (https://itunes.apple.com/in/app/educba-learning-app/id1341654580?mt=8)

Android (https://play.google.com/store/apps/details?id=com.educba.www)

Resources

Free Courses (https://www.educba.com/software-development/free-courses/?source=footer)

Java Tutorials (https://www.educba.com/software-development/software-development-tutorials/java-tutorial/?source=footer)

Python Tutorials (https://www.educba.com/software-development/software-development-tutorials/python-tutorial/?source=footer)

All Tutorials (https://www.educba.com/software-development/software-development-tutorials/?source=footer)

Certification Courses

All Courses (https://www.educba.com/software-development/courses/?source=footer)

Software Development Course - All in One Bundle (https://www.educba.com/software-development/courses/software-development-course/?source=footer)

Become a Python Developer (https://www.educba.com/software-development/courses/python-certification-course/?source=footer)

Java Course (https://www.educba.com/software-development/courses/java-course/?source=footer)

(https://www.educba.com/software-development/)

VB.NET Course (https://www.educba.com/software-development/courses/vb-net-course/?source=footer)

PHP Course (https://www.educba.com/software-development/courses/php-course/?source=footer)