Clearing The Input Buffer In C/C++

Difficulty Level : Medium • Last Updated : 30 Oct, 2022

What is a buffer?

A temporary storage area is called a buffer. All standard input and output devices contain an input and output buffer. In standard C/C++, streams are buffered. For example, in the case of standard input, when we press the key on the keyboard, it isn't sent to your program, instead of that, it is sent to the buffer by the operating system, till the time is allotted to that program.

How does it affect Programming?

On various occasions, you may need to clear the unwanted buffer so as to get the next input in the desired container and not in the buffer of the previous variable. For example, in the case of C after encountering "scanf()", if we need to input a character array or character, and in the case of C++, after encountering the "cin" statement, we require to input a character array or a string, we require to clear the input buffer or else the desired input is occupied by a buffer of the previous variable, not by the desired container. On pressing "Enter" (carriage return) on the output screen after the first input, as the buffer of the previous variable was the space for a new container(as we didn't clear it), the program skips the following input of the container.

In the case of C Programming

C

```c
// C Code to explain why not
// clearing the input buffer
// causes undesired outputs
#include <stdio.h>
int main()
{
    char str[80], ch;

    // Scan input from user -
    // GeeksforGeeks for example
    scanf("%s", str);

    // Scan character from user-
    // 'a' for example
    ch = getchar();

    // Printing character array,
    // prints “GeeksforGeeks”)
    printf("%s\n", str);

    // This does not print
    // character 'a'
    printf("%c", ch);

    return 0;
}
```

Input:

```
GeeksforGeeks
a
```

Output:

```
GeeksforGeeks
```

Time Complexity: O(1)

In the case of C++

C++

```cpp
// C++ Code to explain why
// not clearing the input
// buffer causes undesired
```

```
// outputs
#include<iostream>
#include<vector>
using namespace std;

int main()
{
    int a;
    char ch[80];

    // Enter input from user
    // - 4 for example
    cin >> a;

    // Get input from user -
    // "GeeksforGeeks" for example
    cin.getline(ch,80);

    // Prints 4
    cout << a << endl;

    // Printing string : This does
    // not print string
    cout << ch << endl;

    return 0;
}
```

Input:

```
4
GeeksforGeeks
```

Output:

```
4
```

Time Complexity: O(1)

In both of the above codes, the output is not printed as desired. The reason for this is an occupied Buffer. The "\n" character goes remains there in the buffer and is read as the next input.

How can it be resolved?

In the case of C:

1. Using " while ((getchar()) != '\n'); ": Typing "while ((getchar()) != '\n');" reads the buffer characters till the end and discards them(including newline) and using it after the "scanf()" statement clears the input buffer and allows the input in the desired container.

C

```c
// C Code to explain why adding
// "while ( (getchar()) != '\n');"
// after "scanf()" statement
// flushes the input buffer
#include<stdio.h>

int main()
{
    char str[80], ch;

    // scan input from user -
    // GeeksforGeeks for example
    scanf("%s", str);

    // flushes the standard input
    // (clears the input buffer)
    while ((getchar()) != '\n');

    // scan character from user -
    // 'a' for example
    ch = getchar();

    // Printing character array,
    // prints “GeeksforGeeks”)
    printf("%s\n", str);

    // Printing character a: It
    // will print 'a' this time
    printf("%c", ch);

    return 0;
}
```

Input:

```
GeeksforGeeks
a
```

Output:

```
GeeksforGeeks
a
```

Time Complexity: O(n), where n is the size of the string.

2. Using “ fflush(stdin) ”: Typing “fflush(stdin)” after “scanf()” statement, also clears the input buffer but generally it’s use is avoided and is termed to be “undefined” for input

stream as per the C++11 standards.

In the case of C++:

1. Using " cin.ignore(numeric_limits::max(),'\n'); " :- Typing "cin.ignore(numeric_limits::max(),'\n');" after the "cin" statement discards everything in the input stream including the newline.

C++

```
// C++ Code to explain how
// "cin.ignore(numeric_limits
// <streamsize>::max(),'\n');"
// discards the input buffer
#include <iostream>

// for <streamsize>
#include <ios>

// for numeric_limits
#include <limits>
using namespace std;

int main()
{
    int a;
    char str[80];

    // Enter input from user
    // - 4 for example
    cin >> a;

    // discards the input buffer
    cin.ignore(numeric_limits<streamsize>::max(), '\n');

    // Get input from user -
    // GeeksforGeeks for example
    cin.getline(str, 80);

    // Prints 4
    cout << a << endl;

    // Printing string : This
    // will print string now
    cout << str << endl;

    return 0;
}
```

Input:

```
4
GeeksforGeeks
```

Output:

```
4
GeeksforGeeks
```

Time Complexity: O(1)

2. Using " cin.sync() ": Typing "cin.sync()" after the "cin" statement discards all that is left in the buffer. Though "cin.sync()" **does not work** in all implementations (According to C++11 and above standards).

C++

```
// C++ Code to explain how " cin.sync();"
// discards the input buffer
#include <ios>
#include <iostream>
#include <limits>

using namespace std;

int main()
{
    int a;
    char str[80];

    // Enter input from user
    // - 4 for example
    cin >> a;

    // Discards the input buffer
    cin.sync();

    // Get input from user -
    // GeeksforGeeks for example
    cin.getline(str, 80);

    // Prints 4
    cout << a << endl;

    // Printing string - this
    // will print string now
    cout << str << endl;

    return 0;
}
```

Input:

```
4
GeeksforGeeks
```

Output:

```
4
```

Time Complexity: O(1)

3. Using " cin >> ws ": Typing "cin>>ws" after "cin" statement tells the compiler to ignore buffer and also to discard all the whitespaces before the actual content of string or character array.

C++

```
// C++ Code to explain how "cin >> ws"
// discards the input buffer along with
// initial white spaces of string

#include<iostream>
#include<vector>
using namespace std;

int main()
{
    int a;
    string s;

    // Enter input from user -
    // 4 for example
    cin >> a;

    // Discards the input buffer and
    // initial white spaces of string
    cin >> ws;

    // Get input from user -
    // GeeksforGeeks for example
    getline(cin, s);

    // Prints 4 and GeeksforGeeks :
    // will execute print a and s
    cout << a << endl;
    cout << s << endl;

    return 0;
}
```

Input:

```
4
GeeksforGeeks
```

Output:

```
4
GeeksforGeeks
```

Time Complexity: O(1)

4.Using " fflush(stdin) ": Typing "fflush(stdin)" after taking the input stream by "cin" statement also clears the input buffer by prompting the '\n' to the nextline literal but generally it is avoided as it is only defined for the C++ versions below 11 standards.

C++

```cpp
// C++ Code to explain working of "flush(stdin);"
// discards the input buffer
#include <cstdio> //fflush(stdin) is available in cstdio header files
#include <ios>
#include <iostream>
using namespace std;

int main()
{
    int value;
    string letters;

    // Enter an integer input from user
    // 98 for example
    cin >> value;

    // Discards the input buffer
    fflush(stdin);

    // Get input from user -
    // GeeksforGeeks for example
    getline(cin, letters);

    // Prints 98
    cout << value << endl;

    // Printing user entered string - this
    // will print string now
    cout << letters << endl;

    return 0;
}
```

```
// This code is contributed by im_impossible_ksv
```

Input:

```
369
geeksforgeeks
```

Output:

```
369
geeksforgeeks
```

Time Complexity: O(1)

This article is contributed by **Manjeet Singh**. If you like GeeksforGeeks and would like to contribute, you can also write an article and mail your article to review-team@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.
Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

290

Related Articles

1. main Function in C
2. printf in C
3. C Program to Implement Max Heap
4. C Program to Implement Min Heap
5. C++ Error - Does not name a type
6. Execution Policy of STL Algorithms in Modern C++
7. C++ Program To Print Pyramid Patterns
8. Jagged Arrays in C++