Caution when reading char with scanf (C)


gsamaras.wordpress.com/code/caution-when-reading-char-with-scanf-c

February 15, 2013

Quite often I see the following problem in people's code (as a matter of fact, aliens don't do that mistake :p), when trying to input a character again and again with scanf. Usually, this happens inside a loop, but let's see a sample code without a loop that exposes the problem.

badCscanf.c

```
#include
int main( void )
{
char c;

printf ( "Input No.1\n" );
scanf ( "%c" , &c);
printf ( "c = %c\n" , c);

printf ( "Input No.2\n" );
scanf ( "%c" , &c);
printf ( "c = %c\n" , c);

printf ( "Input No.3\n" );
scanf ( "%c" , &c);
printf ( "c = %c\n" , c);

return 0;
}
```

which gives *output*

```
Input No.1
s
c = s
Input No.2
c =
Input No.3
a
c = a
RUN SUCCESSFUL (total time: 5s)
```

As you see, the input No.2 was skipped. Well not really, not at all. But the intention of the programmer is usually to get three characters from the user, which is what is happening, isn't? Let me explain. Think what you do, when you input. First prompt message arrives,

you type s and then what? You hit enter! Enter is a character!

As a result, first scanf will read the s. Second scanf will read the enter! That's why, the second printf of the value of c leaves just a newline after "c=". Then the third scanf waits for a key press. You input a and then you hit enter. a is been assigned to variable c and enter remains in the stdin buffer, ready to be read by the next scanf. If we had a fourth scanf, then it would read the enter.

However, many would think now that, "Wait a minute... I do this while reading numbers, with %d for example and I had no problem". Correct! Why? Because %d automatically eats whitespaces and special characters. Logical, isn't it? Whitespaces and special characters are characters, not numbers! However, %c has to interpret whitespaces and special characters as inputs, because %c reads characters . Luckily enough, the "fix to scanf to do what you intend it to do" is to leave a space before %c. That way you say to scanf to automatically eat whitespaces and special characters, like enter! So, just change scanf("%c", &c); to scanf(" %c", &c); and you will be just fine. See by yourself in the code bellow:

goodCscanf.c

```
1  #include
2  int main( void )
3  {
4  char c;
5
6  printf ( "Input No.1\n" );
7  scanf ( "%c" , &c);
8  printf ( "c = %c\n" , c);
9
10 printf ( "Input No.2\n" );
11 scanf ( " %c" , &c);
12 printf ( "c = %c\n" , c);
13
14 printf ( "Input No.3\n" );
15 scanf ( " %c" , &c);
16 printf ( "c = %c\n" , c);
17
18 return 0;
19 }
20
21
```

which outputs, what you expect:

```
Input No.1
s
c = s
Input No.2
a
c = a
Input No.3
m
c = m
RUN SUCCESSFUL (total  time : 4s)
```

Notice that in the first scanf, there is no need for a space before %c, since it's the first input function, thus there is no trailing newline to eat, as discussed in the comments section below my post.

I suggest taking a look at my relevant answer in Stackoverflow too.

This code was developed by me, **G. Samaras**. There is also a nice answer in this Stackoverflow question.

Have questions about this code? Comments? Did you find a bug? Let me know!
*Page created by **G. (George) Samaras (DIT)***