SFT 221

Workshop 1

Name:	Meet Hitesh Sonagara
ID:	122208226
Email:	msonagara@myseneca.ca
Section:	ZEE

Authenticity Declaration:
I declare this submission is the result of my own work and has not been shared with any other student or 3rd party content provider. This submitted piece of work is entirely of my own creation.

Test #	1
Test name	Positive Nominal Test Case

Original Code	char s1[] = { "hello, I am meet" };
	char prefix[] = { "he" };
	char suffix[] = { "er" };

Modified Code	N/A
Bug or not?	no
Test Description	By Providing the correct string to verify that the program runs perfrectly as it supposed to be.
Test Results	#VALUE!

Test #	2
Test name	Minimal edge case

Original Code	char s1[] = { "hello, I am Meet" };
	char prefix[] = { " " };
	char suffix[] = { " " };

Modified Code	N/A
Bug or not?	yes
Test Description	By giving empty string input in prefix and suffix to check that the program works properly.
Test Results	Failed because code is not able to handle an empty string it should show an error when the values are empty such as "Empty string not supported" or "Please enter a value, empty string not supported".
Test Results	#VALUE!

Test #	3
Test name	Maximum Edge Case

Original Code	char s1[] = { "hello,I am Meet, how are you..?? I hope You are Good" };
	char prefix[] = { "hello,I am Meet, how are" };
	char suffix[] = { ".?? I hope You are Good" };

Modified Code	
Bug or not?	Yes
Test Description	I have given input of more than the fixed limit of buffer which is 20
Test Results	#VALUE!

Test #	4
Test name	

Original Code	char s1[] = { "" };
	char prefix[] = { "hello" };
	char suffix[] = { "hello" };

Modified Code	N/A
Bug or not?	Yes
Test Description	No input in s1 putting it empty and inputing the same value in pefix and suffix
Test Results	#VALUE!
Test Results	Here it should show error but its is not showing, rather than showing an error it is prints the output without any thing whereas program should stop working.

Bug #	1
Test #	3

Original code

```
int startsWith(const char s[], const char prefix[])
{
    char buf[20];
    int i;

    int sz = strlen(prefix);
    for (i = 0; i < sz; i++)
    {
        buf[i] = s[i];
    }
    buf[sz] = '\0';
    return 0 == strcmp(buf, prefix);
}
int endsWith(const char s[], const char suffix[])
{
    int sz = strlen(suffix);
    int slen = strlen(s);
    return 0 == strcmp(s + slen - sz, suffix);
}
```

Fixed code

```
int startsWith(const char s[], const char prefix[])
{
    int sz = strlen(prefix);
    return strncmp(s, prefix, sz)==0;
}
int endsWith(const char s[], const char suffix[])
{
    int sz = strlen(suffix);
    int slen = strlen(s);
    if (slen < sz) {
        return 0;
    }
    return strncmp(s + slen - sz, suffix, sz)==0;
}
```

Explanations

So I have updated and fixed a very big issuse here by removing the fixed sized or the limit of input string and buffer starts overflowing due to this. To Acoid the, I simple removed the buffer of any fixed sized and there will be not needed to separetly allocate buffer.

A reflection where you consider whether testing or inspection identified more bugs in this case. State why you think one way worked better than the other. How could you improve the technique that worked less well?

Here first of all wht I think is that testing help me a lot in finding small small errors and other issues in different different conditions, because of the tesing which is more efficient in detecting the fault in which there was a cases where there was a empty input string because testing allows me to clearly review this type of edge cases. It also assisted me in detecting a runtime errors and observable issues in the code. I will push myself to think more and more on this and about potential outputs.

Did you find it difficult to find the bugs in this assignment? If not, what helped find them quickly? If you did find it difficult, what made finding the bugs so difficult?

To clarify the bugs in this assignment was not difficult, to me it seems pretty easy because the code was very simple and to figure out the bug still exist or not. I have paid attention to the test cases technique in which I have bulit few of edge cases and tested them Which help me to find the errors very quickly. I was familiar with the C programming language and its string manipulation methods, such as strlen and strcmp this things made really easy for me to find the bugs and identify the errors in this code with this method.