Aymane Arfaoui
Github: Aymane-Arfaoui
ID: 923812458
CSC415 Operating Systems

Assignment 2 – Buffer and Structure

Description:
The goal of this assignment is to understand the use of concepts such as structures, pointers, character strings, enumerated types, bitmap fields, and buffering data into blocks

Approach:
I first need to do research on the different kind of structures necessary for this assignment.
What is a structure and how does it work ?

C structures are a way of grouping several related variables in one place.
https://www.w3schools.com/c/c_structs.php

Creating a Struct and using it:

```
struct myStructure {
  int myNum;
  char myLetter;
};

int main(){
      struct myStruct s1;
      s1.myNum =  13;
      s1.myLetter = 'B';
}
```

Now we need to understand what exactly is typedef struct:
"The **typedef** is a keyword that is used to provide existing data types with a new name. The C typedef keyword is used to redefine the name of already existing data types."

Now an understanding of malloc is required.
Malloc is a memory allocation method that allows us to change the size of the memory at run time.

We need to import stdlib.h in order to use malloc.

Typedef can also be used with a structure:
https://www.geeksforgeeks.org/typedef-in-c/

Typedef is used with structures. A new data type can be created and used to define a structure variable.

How Strings In C work:
In C, strings are an array of characters ending with what we call a null terminator that indicates that this is the end of the string. It is thus important to add a + 1 when allocating memory using malloc for that null terminator.

What is ENUM ?:

Enum is a special type that represents a group of constants.
https://www.w3schools.com/c/c_enums.php

Issues and Resolutions:

1.One of my issues is related to step 6 that involves getting a series of C strings by using the function getNext(). I don't currently know how to execute that step.

My approach is to understand how to use memcpy as this is what is used to fill BLOCK_SIZE

The C library function **void *memcpy(void *dest, const void *src, size_t n)** copies **n** characters from memory area **src** to memory area **dest**.

https://www.tutorialspoint.com/c_standard_library/c_function_memcpy.htm

2. Another issue that I need to resolve is the following: The teacher asks us for malloc which is used for dynamic memory allocation when we know the size of BLOCK_SIZE. The question is why ?

The reason is that even if the size was known, it does not mean that we will fully use all the space allocated. Malloc can then be used to lessen the space required.
https://www.geeksforgeeks.org/dynamic-memory-allocation-in-c-using-malloc-calloc-free-and-realloc/

3. I built my program but I am encountering issues. I see that there is some potential problems regarding the getNext() function. I get the following error message:

munmap_chunk(): invalid pointer
Aborted (core dumped)
make: *** [Makefile:70: run] Error 134

so I am thinking, maybe i need to point to the first letter of the string of getNext() before calling the function.

I need to understand this function call:

const char * getNext(void);

This declaration means the following:
1) the const keyword means that the value that the pointer is pointing to is constant, that is read-only and cannot be changed.
2) we use “char”, to specify that the function returns a pointer to a character.
3) * the star means we are returning an address
4) getNext() is the name of the function and we say void to mean that the function takes no parameter

Solution: I understood my mistake. I did not update the value of nextString by assigning it to getNext(). Here is the line of code that helped that i placed inside my while -loop .

nextString = getNext().

Now the issue that I have is a segmentation fault. What exactly is a segmentation fault ?

Segmentation fault is a runtime error that occurs when the program tries to access a region of the memory that is not allowed.

I have fixed the segmentation fault. I believe it was inside my while loop again and it was due to the fact that I kept creating a new variable inside the while loop

Problem #3:
Check failed:
assumption #1: wrong handling of the free and malloc.

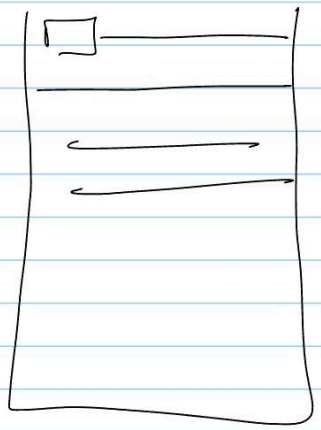
I need to better understand how to use buffers in C.

Problem #4: My logic for approaching the buffer problem was wrong. I went and saw Professor Bierman and I decided to draw the whole approach to fix my logic and make sure I am correctly approaching the problem.

I decided to include a portion of my drawings to show the logic of my code. It helped me a lot to figure out how to properly copy items to the buffer.

if remainder > strlength

1



Before:

strlength = 100

BufferUsed = 0

Remainder 256

After:

BufferUsed = 100

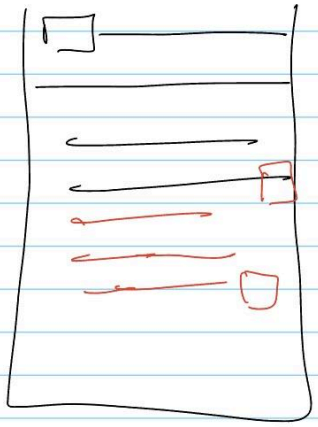
Remainder: 156.

memcpy (buffer+BufferUsed, strignext, strlength)

BufferUsed += strlength

remainder = Block-size − BufferUsed

2



Before:

strlength = 100

BufferUsed = 100

Remainder = 156

After:

BufferUsed = 200

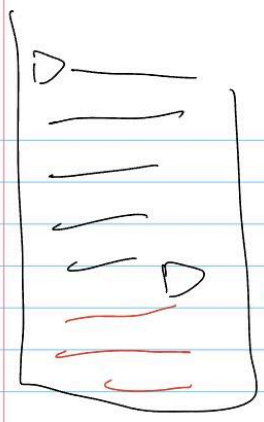
Remainde = 56

memcpy (buffer+Bufferused, strignext, strlength);

BufferUsed += strlength

remainde = BlockSize − Bufferused

(3)

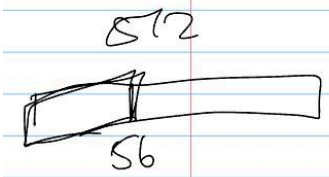


While strength > remainder

Before
strength = 512
BufferUsed = 200
Remainder = 56

memcpy(Buffer+BufferUsed, stringnext, remainder)
BufferUsed = 0
remainder = Blocksize - BufferUsed
Commit(Buffer)
stringoffset = remainder;

memcpy(Buffer+BufferUsed, nextstring+offset, remainder
Commit(Buffer)



BufferUsed = 0
remainder = Blocksize - BufferUsed;
offset += Remainder

After
BufferUsed = 256
Remainder = 0
StringLeft = 512 - 56 = 456

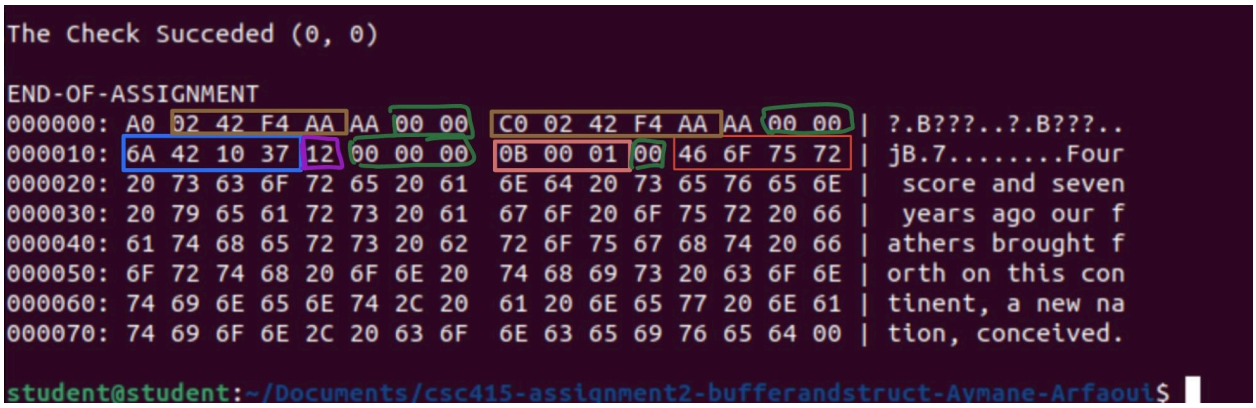
(4)

Before
BufferUsed = 0
Remainder = 256
StringLeft = 456

After
BufferUsed = 256
Remainder = 0

StringLeft = 456 - 256
= 200

Analysis:



The sequences in brown are the address location of the first name and last name. The reason for that is because we used malloc which is dynamic memory allocation. and therefore the memory is not always the same. This is how I understood why memory addresses keep changing.

I used the following code to determine the memory location and value at the memory location:

printf("address of p1.firstName: %p\n", (void*)&(p1.firstName));
printf("Value stored at p1.firstName: %s\n\n", p1.firstName);

The part in red is the beginning of the text. The first segment (red box) from 00001C to 0001F is the hexadecimal code for: Four. is 46 6F 75 72. That part is correct. This is also where the text starts.

46 : 6*(16^0) + 4*(16^1) = 70 which corresponds to: F in the ASCII Table
6F: 6*16 + (15*1) = 111 which corresponds to: o in the ASCII Table
75: 7*16 + (5*1) = 117 which corresponds to: u in the ASCII Table
72: 7*16 + (2*1) = 114 which corresponds to: r in the ASCII Table

The part boxed in blue is the HEX code for the student id (923812458). Only it is read from right to left: 37 10 42 6A. 000013 to 000010

37 10 42 6A = $(3 \times 16^7) + (7 \times 16^6) + (1 \times 16^5) + (0 \times 16^4) + (4 \times 16^3) + (2 \times 16^2) + (6 \times 16^1) + (10 \times 16^0)$ = (923812458) (base 10) which corresponds to the student ID.

The sequence in pink corresponds to the languages known which is obtained through the addition of 1 (KNOWLEDGE_OF_C) +2 (KNOWLEDGE_OF_JAVA) + 8 (KNOWLEDGE_OF_PYTHON) + KNOWLEDGE_OF_INTEL_ASSEMBLER (65536). which gets us the value of 65 547 which

corresponds to the value of 1000B in hex. The sequence in pink shows that hex value in the reverse order that is starting from 00001A to 00008.

The value 12 in purple however at (position 000014) represents the hex value for 18 which using the enum (in the C code) corresponds to SOPHMORE.

The values highlighted in green represent null characters (00)
The values A0 at the very beginning (000000) is a new line and AA (position 000005 and 00000D) has no particular meaning in this case

The part after but including 000020 is the message.

Screenshot of compilation:

```
student@student:~/Documents/csc415-assignment2-bufferandstruct-Aymane-Arfaoui$ make
gcc -c -o arfaoui_aymane_HW2_main.o arfaoui_aymane_HW2_main.c -g -I.
gcc -o arfaoui_aymane_HW2_main arfaoui_aymane_HW2_main.o assignment2M1.o -g -I.
student@student:~/Documents/csc415-assignment2-bufferandstruct-Aymane-Arfaoui$
```

Screen shot(s) of the execution of the program:

```
student@student:~/Documents/csc415-assignment2-bufferandstruct-Aymane-Arfaoui$ make run
./arfaoui_aymane_HW2_main aymane arfaoui "Four score and seven years ago our fathers brought forth on t
his continent, a new nation, conceived in Liberty, and dedicated to the proposition that all men are cr
eated equal."


0
---------------------------------- CHECK ----------------------------------
Running the check for aymane arfaoui
Name check is 0 by 0
Student ID: 923812458, Grade Level: Sophmore
Languages: 65547
Message:
Four score and seven years ago our fathers brought forth on this continent, a new nation, conceived

The Check Succeded (0, 0)

END-OF-ASSIGNMENT
000000: A0 D2 9A 19 AB AA 00 00  C0 D2 9A 19 AB AA 00 00 | ???.??..???.??..
000010: 6A 42 10 37 12 00 00 00  0B 00 01 00 46 6F 75 72 | jB.7........Four
000020: 20 73 63 6F 72 65 20 61  6E 64 20 73 65 76 65 6E |  score and seven
000030: 20 79 65 61 72 73 20 61  67 6F 20 6F 75 72 20 66 |  years ago our f
000040: 61 74 68 65 72 73 20 62  72 6F 75 67 68 74 20 66 | athers brought f
000050: 6F 72 74 68 20 6F 6E 20  74 68 69 73 20 63 6F 6E | orth on this con
000060: 74 69 6E 65 6E 74 2C 20  61 20 6E 65 77 20 6E 61 | tinent, a new na
000070: 74 69 6F 6E 2C 20 63 6F  6E 63 65 69 76 65 64 00 | tion, conceived.

student@student:~/Documents/csc415-assignment2-bufferandstruct-Aymane-Arfaoui$
```