CS133C PROGRAMMING IN C

Fall 2022

Week 5 Assignment

Due Tuesday 11/01/22

### How to submit the homework for week 5:

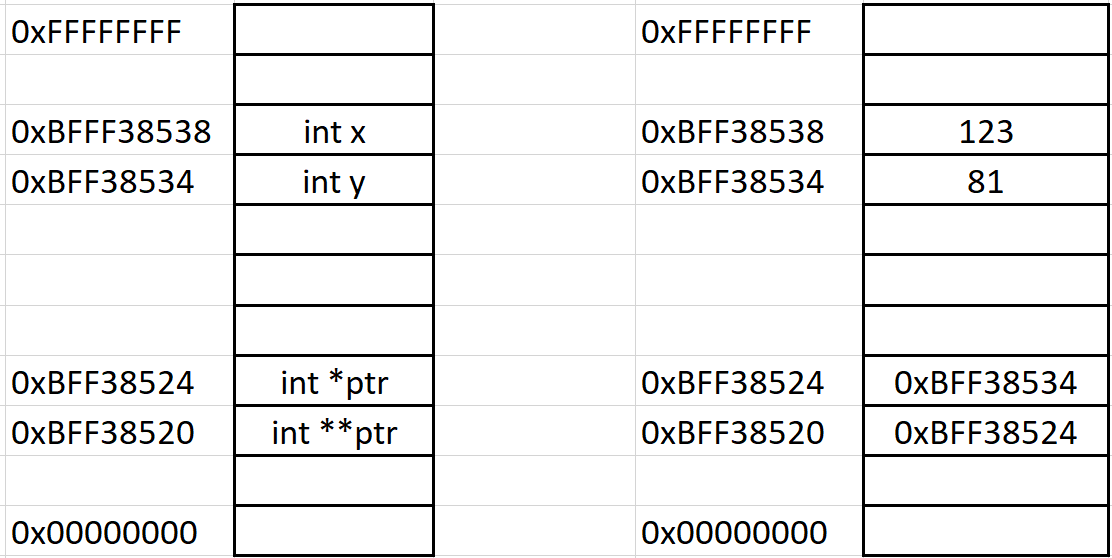
1. For week 5, you will need answer some questions in this Word document, and you will also need to make one program for Question 4b.
2. Please write your answer for question 1 through question 4a in this document and save this document as a new **.docx** file or a **pdf** file named **cs133c\_week05\_your\_initials**.
3. Make one C program named **sort.c**. At the top of the program, please write your **name**, student **ID**, and **date** completed.
4. Save those 2 files (your word/pdf document with answers, and **sort.c**) in a folder named **cs133c\_week05\_your\_initials** under your **…/cs133c/week05** directory.
5. Compress the folder and submit the zip folder, **cs133c\_week05\_your\_initials**, to Moodle.

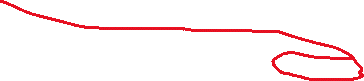
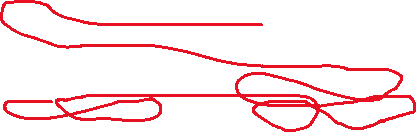
### Question 1 (5 points)

1. A quick review -- if **ptr** is a pointer, then:

* **&ptr** will access the address of a variable with the address operator.
* **ptr** will access a pointer's value by accessing it directly (just like a normal variable).
* **\*ptr** will access the value that the pointer is pointing at (rather than the pointer’s value).
* **%p** as a format specifier that print an address in hexadecimal format.

The two diagrams below are referring to the same thing. The **ptr** and **pptr** in the left diagram are pointers, and the diagram on the right shows the values held in pointers **p** and **pp**.





As we can observe:

* **\*ptr** evaluates to **81**
* **ptr** evaluates to **0xBFF38534**
* **x** evaluates to **123**
* **y** evaluates to **81**

Let’s say we have an **int \*\*pptr** that is located at **0xBFF38520**.

**Q:** What does pptr evaluate to? How about \*pptr? What about \*\*pptr?

**Answer:**

**PPTR = 0xBFF38534**



**\*PPTR = 81**

**\*\*PPTR = 81**

1. What is the value of **\*ptr** and of **\*(ptr + 2)** in the cases below?

int \*ptr;

int array\_one[2][2] = {10, 5, 6};

ptr = &array\_one [0][0];

**Answer:**

**\*PTR = 10**

**12**

**//Not sure I quite understand the arithmetic for pointers**

int \*ptr;

int array\_two[2][2] = { {81}, {32, 59} };

ptr = &array\_two[0][0];

**Answer:**

**\*PTR = 81**

**83**

### Question 2 (3 points)

Choose one or more of the following memory segments where the data could be located:

**code**, **heap**, and/or **stack**.

1. local variables

**Answer:**

**Code**

1. result of malloc() function

**Answer:**

**Heap**

1. executable instructions

**Answer:**

**Code**

**Stack**

### Questions 3 (4 points)

How to allocate memory on the heap in the following scenarios.

1. An array named **arr** **of n integers**. (one line of code)

**Answer:**

**int \*arr = malloc(n\*sizeof(int));**

1. A string named **str containing n characters**. (one line of code)

**hint:** do not forget the null terminator.

**Answer:**

**char \*str = malloc(n\*sizeof(char)){return NULL;}**

### Question 4 (8 points)

Working with pointers. Implement the following functions so that they work as described.

1. Write a **function** that returns the number of bytes in a string. Do not use strlen().

**hint:** the function header can be **int str\_len(const char \*str)**

**Answer:**

#include<stdio.h>

int main()

{

    char word[5] = {'h', 'e', 'l', 'l', 'o'};

    printf("sizeof(char) = %d\n\n", sizeof(word));

    return 0;

}

1. Write a program named **sort.c** which let user input three integers, and then print out the three user-input integers from the smallest to the largest. Please make use of pointers in your program. (this one will be sort.c source file)

Text

Description automatically generated

I tested w/diff numbers