**CSCD 240**

NOTE: Your answers, for all problems, will be saved in a file named cscd240Lab8.pdf for all problems

NOTE: Your C file will be named cscd240Lab8.c

1. Type in, compile and execute the following code.

#include <stdio.h>

#include <stdlib.h>

#define R 5

#define C 4

int main(int argc, const char \* argv[])

{

int i, j, walk = 0;

int \*\*grades = (int \*\*)malloc(R \* sizeof(int \*)); **//Assume that value of this entire expression is 0x7fb322403930**

// allocate memory for each row for test purpose, otherwise we should do the following in a loop

grades[0] = (int \*) malloc( C \* sizeof(int) ); **//Assume that value of this entire expression is 0x7fb3224000e0**

grades[1] = (int \*) malloc( C \* sizeof(int) ); **//Assume that value of this entire expression is 0x7fb322401f50**

grades[2] = (int \*) malloc( C \* sizeof(int) );

grades[3] = (int \*) malloc( C \* sizeof(int) );

grades[4] = (int \*) malloc( C \* sizeof(int) );

for( i = 0; i < R; i ++)

{

for(j = 0; j < C; j ++)

{

grades[i][j] = walk ++;

}

}

int \*\*pptr = grades;

printf("-1: pptr= %p\n", pptr);

printf("-1: &pptr[0] = %p\n", &pptr[0] );

printf("-1: pptr+1= %p\n", pptr + 1);

printf("-1: pptr+2= %p\n", pptr + 2);

printf("\n\n");

printf("0: pptr[0]= %p\n", pptr[0]);

printf("0: \*pptr= %p\n", \*pptr);

printf("0: &pptr[0][0]= %p\n", &pptr[0][0]);

printf("\n\n");

printf("1: pptr[1]= %p\n", pptr[1]);

printf("1: \*(pptr + 1)= %p\n", \*(pptr + 1));

printf("1: &pptr[1][0] = %p\n", &pptr[1][0] );

printf("\n\n");

printf("2: \*pptr + 1 = %p\n", \*pptr + 1);

printf("2: \*(pptr+0) + 1 = %p\n", \*(pptr + 0) + 1);

printf("2: &pptr[0][1] = %p\n", &pptr[0][1] );

printf("\n\n");

printf("3: \*(pptr[1] + 1) = %d\n", \*(pptr[1] + 1) );

printf("3: \*( \*(pptr + 1) + 1) = %d\n", \*( \* (pptr + 1) + 1) );

printf("3: pptr[1][1] = %d\n", pptr[1][1] );

return 0;

} //end of main

This code will provide a base address for grades and pptr.

**Assume you get a 64-bit machine, i.e. an address takes 8 bytes long. Also, each integer on your machine takes 4 bytes long.** Use the base address provided from the comments to the right of malloc() function. Based on the code above, create an educated guess that clearly outlines what you believe will print out as each line of code above is executed. In your explanation clearly **explain what is happening**, don't just give memory addresses or values. If you only provide memory addresses or values you will receive **0 points** for this problem. Your guesses will be clearly labeled in the PDF file. You must provide the line of code and then the explanation.

**TO TURN IN:**

A zip file containing:

* Your PDF file
* Your C file

You better know the naming scheme.