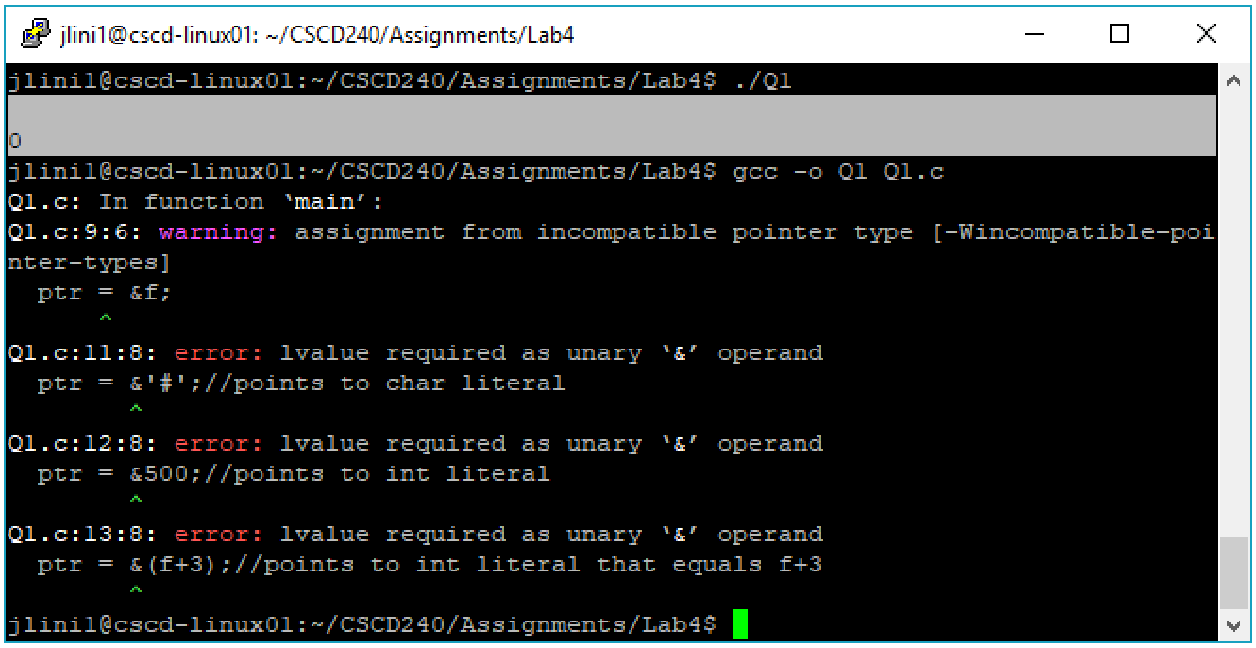
Joshua Lini 2/10/19

Lab 4: Playing around with Pointers

1. Identify and explain which of the following expressions are valid and which are not valid as shown with an arrow. 5 pts

```
char c;
char *ptr;
int f;
ptr = &c;          ←
ptr = &f;          ←
ptr = &'#';        ←
ptr = &500;        ←
ptr = &(f+3);      ←
```



jlini1@cscd-linux01: ~/CSCD240/Assignments/Lab4

```
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$ ./Q1

0
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$ gcc -o Q1 Q1.c
Q1.c: In function ‘main’:
Q1.c:9:6: warning: assignment from incompatible pointer type [-Wincompatible-poi
nter-types]
  ptr = &f;
      ^
Q1.c:11:8: error: lvalue required as unary ‘&’ operand
  ptr = &'#';//points to char literal
        ^
Q1.c:12:8: error: lvalue required as unary ‘&’ operand
  ptr = &500;//points to int literal
        ^
Q1.c:13:8: error: lvalue required as unary ‘&’ operand
  ptr = &(f+3);//points to int literal that equals f+3
        ^
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$
```

Highlighted lines is output from code with the 3 invalid ptr assignments commented out

The errors are from the fact that the 3 bottom ptr assignments are invalid due to being pointed to literal char and int values that aren’t stored in a variable.

Joshua Lini 2/10/19

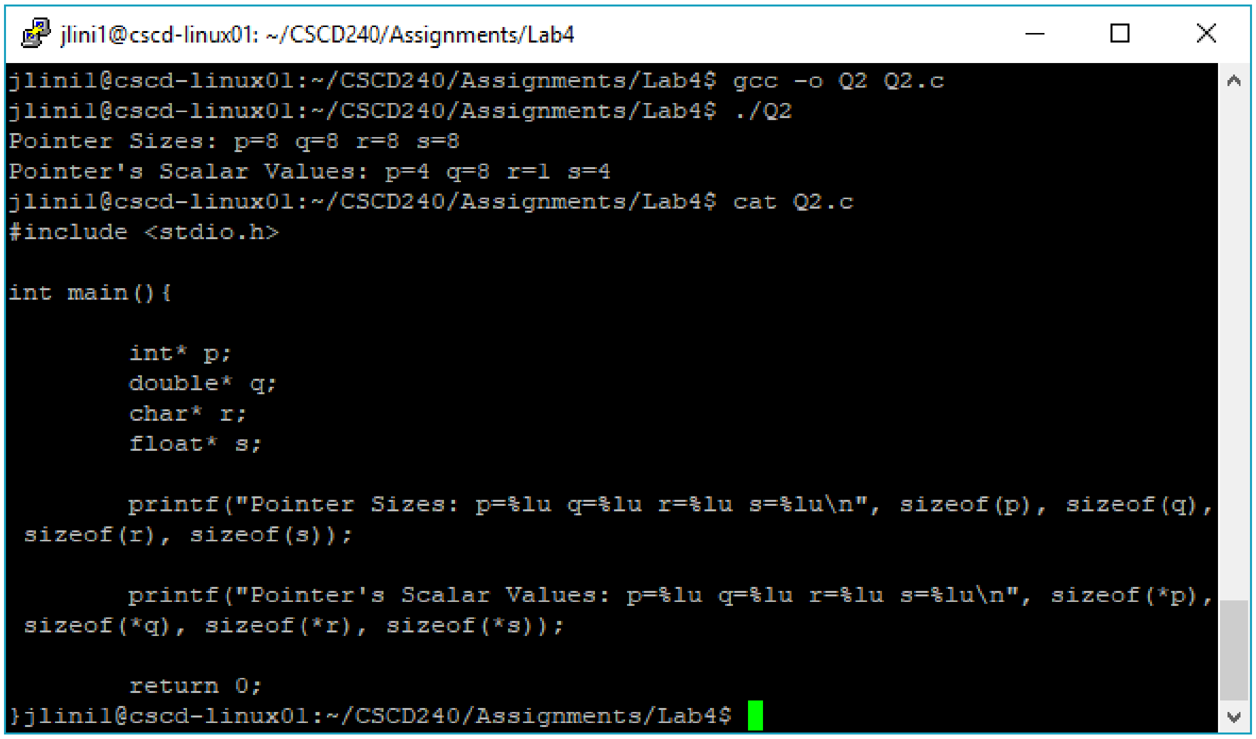
2. **What will be the size of the following pointers? What are their scalar values? Explain with screen shots. 4 pts**

int * p;

double *q;

char* r;

float* s;



jlini1@cscd-linux01: ~/CSCD240/Assignments/Lab4

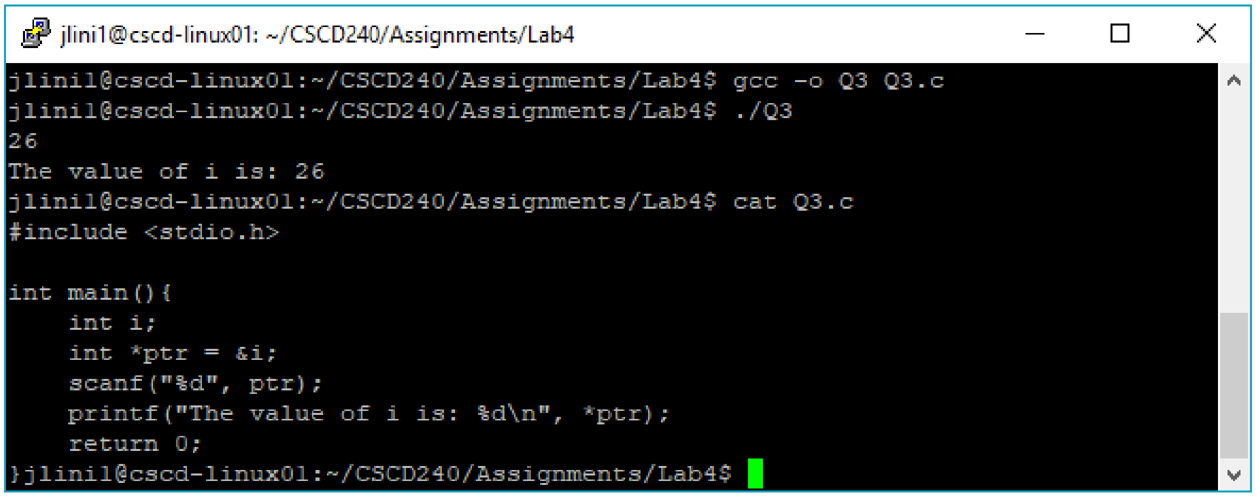
```
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$ gcc -o Q2 Q2.c
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$ ./Q2
Pointer Sizes: p=8 q=8 r=8 s=8
Pointer's Scalar Values: p=4 q=8 r=1 s=4
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$ cat Q2.c
#include <stdio.h>

int main(){

        int* p;
        double* q;
        char* r;
        float* s;

        printf("Pointer Sizes: p=%lu q=%lu r=%lu s=%lu\n", sizeof(p), sizeof(q),
 sizeof(r), sizeof(s));

        printf("Pointer's Scalar Values: p=%lu q=%lu r=%lu s=%lu\n", sizeof(*p),
 sizeof(*q), sizeof(*r), sizeof(*s));

        return 0;
}jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$
```

Joshua Lini 2/10/19

3. **What is wrong with the following program? Please explain. How will you fix it? 1 pt**

```
#include <stdio.h>
int main(){
        int i;
        int *ptr = &i;
        scanf("%d", &ptr);
        printf("The value of i is: %d\n", *ptr);
        return 0;
}
```

The program is asking for a new address to assign to ptr, assuming the program is supposed to set the value of i using ptr, this is the revised code:



jlini1@cscd-linux01: ~/CSCD240/Assignments/Lab4

```
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$ gcc -o Q3 Q3.c
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$ ./Q3
26
The value of i is: 26
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$ cat Q3.c
#include <stdio.h>

int main(){
    int i;
    int *ptr = &i;
    scanf("%d", ptr);
    printf("The value of i is: %d\n", *ptr);
    return 0;
}jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$
```

4. **What will be the values of the variables in the lines marked with arrows.**

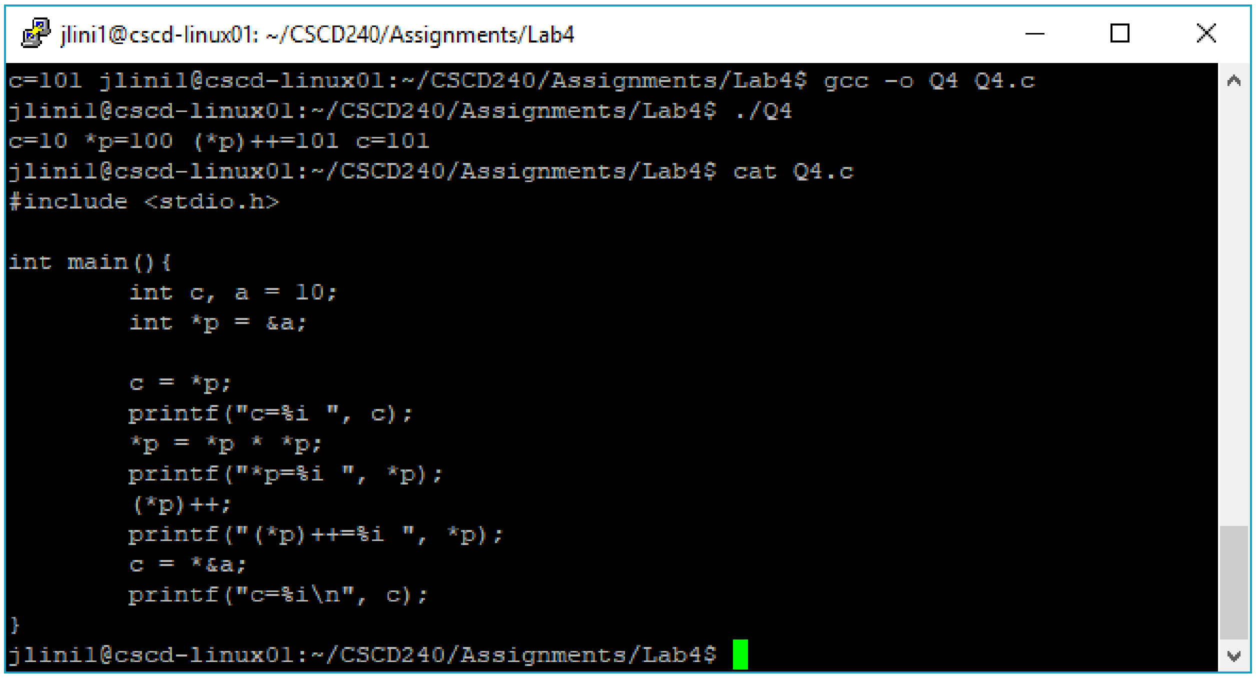
4 pts

```
int c, a = 10;
int *p = &a;
c = *p;          ← 10
*p = *p * *p;    ← 100
(*p)++;          ← ~~101~~After more explicit instructions in-class my answer is 100
```

c = *&a; ← 101

code is still original, but per in class instructions I need to combine the operations and printf statements on to one line.

To make the question more clear, "(*p)++;" should be "printf((*p)++);" and the question should ask us what it will print. The problem is that doesn't hold up for c = *p; because printf(c = *p); obviously will not work. The questions just need to have more explanation and be less ambiguous, I am not the only one with this issue and it has happened in previous labs.


jlini1@cscd-linux01: ~/CSCD240/Assignments/Lab4

```
c=101 jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$ gcc -o Q4 Q4.c
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$ ./Q4
c=10 *p=100 (*p)++=101 c=101
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$ cat Q4.c
#include <stdio.h>

int main(){
        int c, a = 10;
        int *p = &a;

        c = *p;
        printf("c=%i ", c);
        *p = *p * *p;
        printf("*p=%i ", *p);
        (*p)++;
        printf("(*p)++=%i ", *p);
        c = *&a;
        printf("c=%i\n", c);
}
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$
```

5. **Consider the following piece of code. What will be printed by *ptr and ptr after the lines shown with an arrow? Explain with screenshots.**

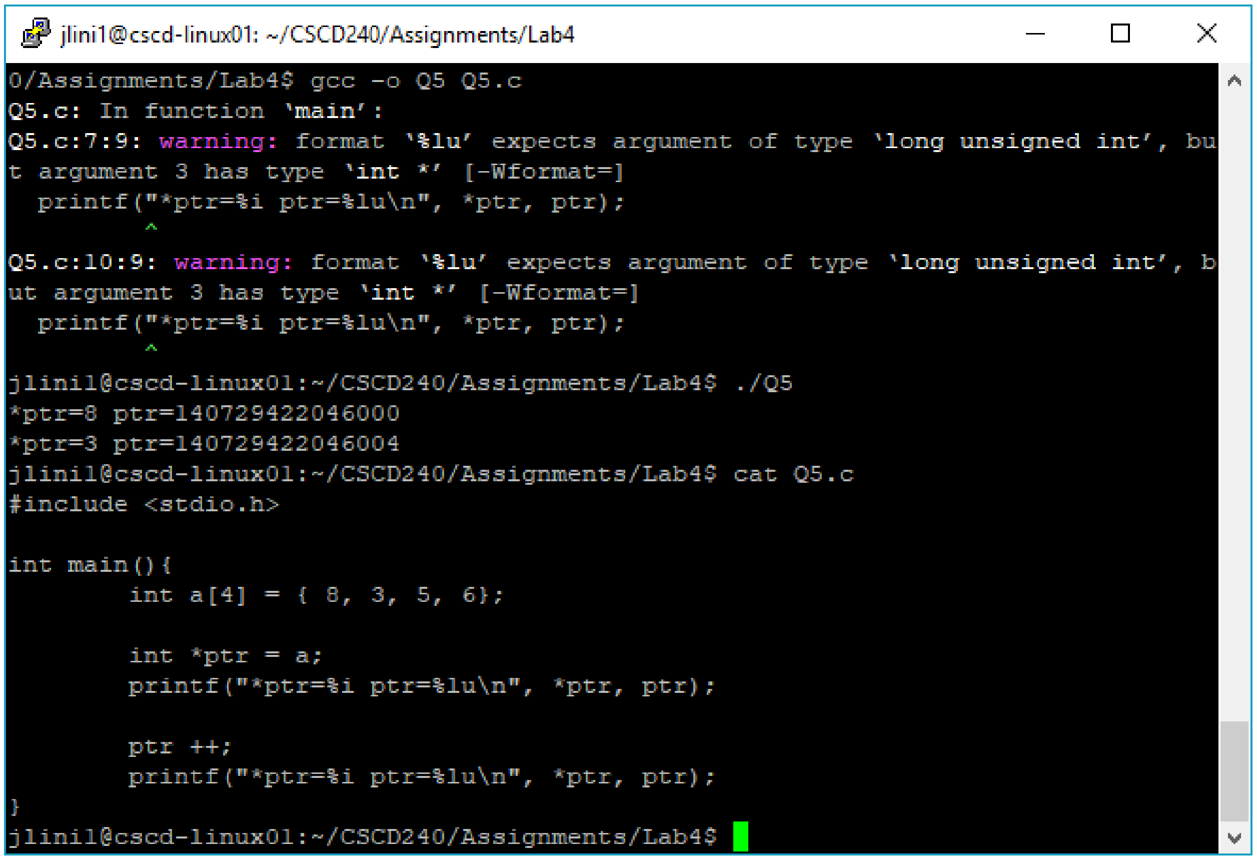
int a[4] = { 8, 3, 5, 6}; **4 pts**

int *ptr = a; ← **8**, memory address of first element

ptr ++; ← 3, **memory address of second element**

I underlined and bolded the answers I think you were looking for, but question was unclear

Joshua Lini 2/10/19



```
0/Assignments/Lab4$ gcc -o Q5 Q5.c
Q5.c: In function ‘main’:
Q5.c:7:9: warning: format ‘%lu’ expects argument of type ‘long unsigned int’, bu
t argument 3 has type ‘int *’ [-Wformat=]
  printf("*ptr=%i ptr=%lu\n", *ptr, ptr);
         ^
Q5.c:10:9: warning: format ‘%lu’ expects argument of type ‘long unsigned int’, b
ut argument 3 has type ‘int *’ [-Wformat=]
  printf("*ptr=%i ptr=%lu\n", *ptr, ptr);
         ^
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$ ./Q5
*ptr=8 ptr=140729422046000
*ptr=3 ptr=140729422046004
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$ cat Q5.c
#include <stdio.h>

int main(){
        int a[4] = { 8, 3, 5, 6};

        int *ptr = a;
        printf("*ptr=%i ptr=%lu\n", *ptr, ptr);

        ptr ++;
        printf("*ptr=%i ptr=%lu\n", *ptr, ptr);
}
jlini1@cscd-linux01:~/CSCD240/Assignments/Lab4$
```

6. What is the difference between the following two declarations? 2 pts

char array[] = “Hello World”;

char array[] creates an editable array of char’s with said text

char *array = “Hello World”;

char *array creates a pointer to ‘H’ and “ello World” is stored character by character in contiguous memory after ‘H’. You shouldn’t try editing this one.

Submission:

A pdf file containing answers to the questions and output capture wherever necessary.
Name your file with your last name first letter of your first name Lab4.pdf (ex: yasminsLab4.pdf).

Submission deadline is: 11:59 pm, Monday, February 11. No late submission will be considered.