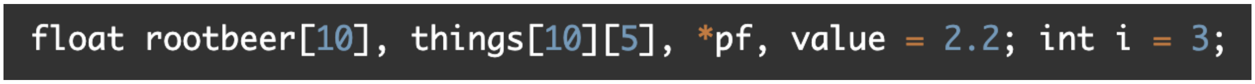
Assignment 4

Task 1: Suppose you have these declarations:



```c
float rootbeer[10], things[10][5], *pf, value = 2.2; int i = 3;
```

Identify each of the following statements as valid or invalid:

Statements	Valid OR Invalid
rootbeer [5] = value;	Valid
scanf ("%f", &rootbeer);	Invalid
rootbeer = value;	Invalid
printf("%f", rootbeer);	Invalid
things[2][2] = rootbeer[2];	Valid
things[3] = rootbeer;	Invalid
pf = value;	Invalid
pf = rootbeer;	Valid

Task 2: Write a function called addtwo that sets each element in an array to the sum of the corresponding elements in two other arrays. That is, if array 1 has the values 1, 2, 3, and 4 and array 2 has the values 10, 20, 30, and 40, the function assigns array 3 the values 11, 22, 33, and 44. The function should take three array names and an array size as arguments. Test the function in a simple program.

Paste your function code in the box below (addtwo function)

```c
void addtwo(int  *array1, int *array2, int *array3, size_t
size)
{
    for (int x = 0; x < size; x++)
    {
        array3[x] = array1[x] + array2[x];
    }
}
```

Paste your main() function code in the box below

```
#include <stdio.h>


int main()
{
    int array1[] = {1,2,3,4,5};
    int array2[] = {1,2,3,4,5};
    int array3[5] = {0};
    addtwo(array1, array2, array3, 5);
    for (int x = 0; x < 5; x++)
    {
        printf("%d\n", array3[x]);
    }
    return 0;
}
```

Task 3: Write a function called mymax that returns the index of the largest value stored in an array-of- double. Test the function in a simple program.

Paste your function code in the box below (mymax function)

```

void mymax(double *array1, size_t length, int *index)
{
    double max = array1[0];
    for (int x = 0; x < length; x++)
    {
        if (array1[x] > max)
        {
            max = array1[x];
            *index = x;
        }
    }
}
```

Paste your main() function code in the box below

```
#include <stdio.h>


int main()
{
    double array1[] = {1.0, 2.3, 3.4, 2.4, 2.0};
    int index = 0;

    mymax(array1, sizeof(array1) / sizeof(array1[0]),
&index);
    printf("%d", index);

    return 0;
}
```

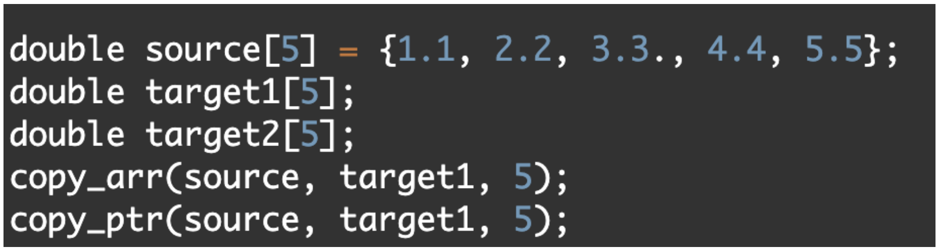
You need to submit two files (task4.c and task5.c) as your answers.

Task 4: Write a program that initializes an array-of-double and then copies the contents of the array into two other arrays. (All three arrays should be declared in the main program.)

To make the first copy, use a function with array notation.

To make the second copy, use a function with pointer notation and pointer incrementing.

Have each function take as arguments the name of the target array and the number of elements to be copied. That is, the function calls would look like this, given the following declarations:



```
double source[5] = {1.1, 2.2, 3.3., 4.4, 5.5};
double target1[5];
double target2[5];
copy_arr(source, target1, 5);
copy_ptr(source, target1, 5);
```

Submit your program as task4.c file to the drop-box

Task 5

Write a program that prompts the user to enter three sets of six double numbers each.
(You may assume the user responds correctly and doesn't enter non-numeric data.)
The program should accomplish all of the following:

a) Store the information in a 3 x 6 array.

b) Compute the average of each set of five values.

c) Compute the average of all the values.

d) Determine the smallest value of the 18 values.

e) Report the results

Each major task should be handled by a separate function using the traditional C approach to handling arrays. Accomplish task "b" by using a function that computes and returns the average of a one-dimensional array; use a loop to call this function three times. The other tasks should take the entire array as an argument, and the functions performing tasks "c" and "d" should return the answer to the calling program.

Submit your program as task5.c file to the drop-box