CS1050 – Prelab 4
Fall 2019

Concepts to Practice

- User-written functions
- math.h

Description

For the prelab assignment, you are to write a function to calculate Power (exponentiation). Your function will be called MyPower. MyPower() will raise a given double type number to a positive integer power. You may **NOT** call any function from math.h such as pow() **in your implementation of MyPower()**.

The main() function in your program should look exactly like this:

```
int main(void)
{
    int i,j;

    for (i=1;i<5;i++)
    {
        for (j=1;j<5;j++)
        {
            DoTest(i,j);
        }
    }
}
```

You will write a DoTest() function which should:

1. Have a prototype of void DoTest(double base, int exponent);
2. Print a message saying that you are testing base (that is, the value passed in as base) to the exponent (that is, the value passed in as exponent) power.
3. Compare the result of raising base to exponent using the MyPower() function that you wrote with the result of raising base to exponent using the pow() function that is prototyped in math.h. If these two results are not equal, print a message indicated the value from the pow() function was expected, but the value from MyPower() was the result.

Functions You Must Write

You may write any functions you wish to implement this program, in **addition** to the following functions. However, you **must** implement the following functions:

- **double MyPower(double base, int exponent)** – This function raises base to the power exponent and returns the result.
- **Void DoTest(double base, int exponent)** – This function prints a message indicating what is being requested. It then tests whether the result using MyPower() is the same as the result using pow(). If the results are the same, it does nothing. If the results are not the same, it prints an error message.
- **int main(void)** – Of course, you need to write a main(). Better yet, I already gave it to you 😊.

Hints

- You should print double numbers with %lf as the formatter.
- Although MyPower() takes an exponent of type int and pow() takes an exponent of type double, the results should be the same for our data. Note that calling pow() with an int is fine because the int is automatically up-cast to a double.

Sample Good Output

```
JimR@JimRArea51:~/CS1050/FS2019/Labs/Lab4$ compile prelab4.c
JimR@JimRArea51:~/CS1050/FS2019/Labs/Lab4$ ./a.out
Testing 1.000000 to the 1 power
Testing 1.000000 to the 2 power
Testing 1.000000 to the 3 power
Testing 1.000000 to the 4 power
Testing 2.000000 to the 1 power
Testing 2.000000 to the 2 power
Testing 2.000000 to the 3 power
Testing 2.000000 to the 4 power
Testing 3.000000 to the 1 power
Testing 3.000000 to the 2 power
Testing 3.000000 to the 3 power
Testing 3.000000 to the 4 power
Testing 4.000000 to the 1 power
Testing 4.000000 to the 2 power
Testing 4.000000 to the 3 power
Testing 4.000000 to the 4 power
```

Sample Output with an Error (you should not have an error, but this shows what it should look like if your MyPower() calculated 3 to the 3rd as -1.32 instead of 27)

```
JimR@JimRArea51:~/CS1050/FS2019/Labs/Lab4$ compile prelab4.c
JimR@JimRArea51:~/CS1050/FS2019/Labs/Lab4$ ./a.out
Testing 1.000000 to the 1 power
Testing 1.000000 to the 2 power
Testing 1.000000 to the 3 power
Testing 1.000000 to the 4 power
Testing 2.000000 to the 1 power
Testing 2.000000 to the 2 power
Testing 2.000000 to the 3 power
Testing 2.000000 to the 4 power
Testing 3.000000 to the 1 power
Testing 3.000000 to the 2 power
Testing 3.000000 to the 3 power
***Error: expected 27.000000, result was -1.320000
Testing 3.000000 to the 4 power
Testing 4.000000 to the 1 power
Testing 4.000000 to the 2 power
Testing 4.000000 to the 3 power
Testing 4.000000 to the 4 power
```