5) In this exercise we want to create and test a function to evaluate \( \sin(x) \) for \( 0 \leq x \leq \frac{\pi}{2} \).

a) Using the remainder estimate for the Taylor series around \( x_0 = 0 \), determine what approximation is needed so that the error will be smaller than \( \frac{1}{2}\text{ulp} \) using IEEE single floating point numbers (respectively: the float data type in C).

b) Implement a function `mysin` in C that calculates \( \sin(x) \) using the approximation from a). You can use the framework code listed after the problem (also available from the course web page).

c) Compare the result of your function with the built-in `sin` function from the C library. How does the error compare with the estimate in a)?

d) Try to minimize the discrepancy between the results produced by your function and those produced by the built-in function. What happens if you use double arithmetic?

Hand in the estimate calculation in a) as well as compilable framework code that includes your `mysin` function.

Framework Code:

```c
#include <math.h>
#include <stdio.h>
float mysin(float x){
    /* your function code here*/
}

int main()
{
    float a,b,x;
    for (x=0;x<1.4;x=x+1./10) {
        a=sin(x);
        b=mysin(x);
        printf("%.10f ",a-b);
    }
    printf("\n");
}
```

Note: To compile this function you have to explicitly specify linking of the math library:

```bash
cc problem5.c -lm
```

6) Consider the following C program. What will its output be?

```c
#include <stdio.h>
int main()
{
    double a,x;
    a=1./5;
    for (x=1;x>-1;x=x-a) {
        if (x==0) {
            printf("reached zero\n");
        }
    }
}
```

Now compile the program and run it. What is the actual output? Explain.

7) Represent \( \frac{1}{10} \) in IEEE single floating point numbers. How many ulp is the rounding error?