/* 
* HW 2 #1 
* This code computes the machine precision for double precision numbers. 
*/

#include <iostream>
#include <iomanip>

using namespace std; // use namespace std, so you can use 'cout' instead of 'std::cout'

int main()
{
    int i = 0;
    double one;
    double eps = 1.0;

    cout << setprecision(20); // setting output precision
    do // do-while loop, execute codes inside {} until 'one' is equal to 1.0 (that means machine cannot distinguish '1.0 + eps' and 1.0)
    {
        eps /= 2.0; // eps = eps/2.0
        one = 1.0 + eps;
        i++;
    }
    while(one != 1.0);
    cout << "eps = " << eps << endl; // output the eps
    return 0; // termination
}

eps = 1.1102230246251565404e-16 // this is the result.

/* 
* HW 2 #2 
* This code asks the user for an integer and then computes the factorial of that number using a recursive function. 
*/

#include <iostream>

using namespace std;

int factorial(int); // declare a function, then you can use it in main();

int main()
{
    int N, fact;
    do // do-while loop
    {
        cout << "Enter an integer you want to compute the factorial of. " << endl;
        cin >> N;
        if(N < 0) // check if the number is legal
            cout << "N must be greater than or equal to zero." << endl;
        } while(N < 0);
    cout << N << "! = " << factorial(N) << endl; // call function 'factorial()' here.
    return 0;
}
int factorial(int a) //compute the factorial here.
{
    if(a > 0)
        return a*factorial(a-1);
    else
        return 1;
}

Enter an integer you want to compute the factorial of.  //output
6
6! = 720

/*
 * HW 2 #3
 * This code computes the Fibonacci series starting from two
 * user-specified starting values of N1, N2. The user also
 * specifies the number of terms to compute after N1 and N2.
 */

#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    int N1, N2, max, temp;
    double ratio;

    cout << setprecision(15); //set precision
    //the following three do-while loop are for getting the user input and judge if they
    are legal
    do
    {
        cout << "Enter the first integer: ";
        cin >> N1;
        if(N1 < 0)
            cout << "N1 must be greater than zero." << endl;
    }
    while(N1 < 0);

    do
    {
        cout << "Enter the second integer: ";
        cin >> N2;
        if(N2 < 0)
            cout << "N2 must be greater than zero." << endl;
        if(N1 == 0 && N2 == 0)
        {
            cout << "Error: N1 and N2 are both zero." << endl;
            N2 = -1;
        }
    }
    while(N2 < 0);

    do
    {
        cout << "Enter the number of terms to compute: ";
        cin >> max;
        if(max < 0)
            cout << "Number of terms must be >= 0" << endl;
        }
while(max < 0); 

for(int i = 0; i < max; i++) //compute Fibonacci series here 
{ 
    temp = N2; 
    N2 += N1; 
    N1 = temp; 
} 
if(N1 == 0) 
{ 
    cout << "Error: Divide by zero. Exiting program." << endl; 
    return(0); //if N1 is zero, exit program 
} 
ratio = (double)N2/N1; 

cout << "Last term: " << N2 << endl; 
cout << "Ratio: " << ratio << endl; 
return 0; 

Enter the first integer: 1 
Enter the second integer: 2 
Enter the number of terms to compute: 15 
Last term: 2584 
Ratio: 1.61803381340013 

/* 
 * HW 2 #4 
 * This code computes the largest and smallest values that 
 * can be stored in a double precision floating point number. 
 */

#include <iostream> 
#include <iomanip> 
#include <cmath> 
using namespace std; 

int main() 
{ 
    double temp, temp2; 
    double dbl_max = 1.0, dbl_min = 1.0; 
    double factor = 8.0; 
    double eps = 1e-15; 
    
    cout << setprecision(15); 
    
    do 
    { 
        temp2 = temp; 
        temp = dbl_max; 
        dbl_max *= 1 + factor; 
        //swap dbl_max and temp with 
        //values from previous iter. 
        if(1/dbl_max == 0) //i.e. if dbl_max == inf 
        { 
            dbl_max = temp; 
            temp = temp2; 
            factor /=2; 
        } 
    }
while(fabs(temp - dbl_max)/dbl_max > eps);
cout << "dbl_max = " << dbl_max << endl;

factor = 8.0;
do {
    temp2 = temp;
temp = dbl_min;
dbl_min /= 1 + factor;
    //swap dbl_min and temp with
    //values from previous iteration
    if(dbl_min <= 0)    //underflow causes the dbl_min -> 0
    {
        dbl_min = temp;
temp = temp2;
factor /= 2;
    }
} while(dbl_min > 0 && fabs(temp - dbl_min)/dbl_min > eps);
cout << "dbl_min = " << dbl_min << endl;
return 0;
}

dbl_max = 1.79769313486232e+308
dbl_min = 4.94065645841247e-324

/*
 * HW 2 #5
 * This code computes the largest value that can be stored
 * in an integer variable.
 */

#include <iostream>
#include <iomanip>
#include <cmath>
using namespace std;

int main()
{
    int i,temp,temp2;
    int int_max = 1;
    double factor = 0.5;

    i = 0;
do {
    i++;
temp2 = temp;
temp = int_max;
    int_max += (int)int_max/factor;

    //swap int_max and temp with
    //values from previous iteration
    if(int_max <= 0)
    {
        int_max = temp;
temp = temp2;
factor *= 2;
    }
}
while(int_max > 0 && abs(temp - int_max) > 0);
cout << "int_max = " << int_max << endl;

return 0;
}

int_max = 2147483647