Вычислить значение функции:

$$f(x,y) = \begin{cases} a * u, & \text{если a <= u, u} < 0, \\ u^2 - a, & \text{если u} > 0, \\ u & \text{в остальных случаях.} \end{cases},$$

При вычислении значения необходимо использовать условную операцию сравнения ?:

Задание 1. Функция

Вычислить значение функции:

$$f(x,y) = \begin{cases} x + y, & \text{если x} > 0, \\ x * y, & \text{если x <= 0, y} < 0, \\ 5 * x & \text{в остальных случаях.} \end{cases},$$

При вычислении значения необходимо использовать условную операцию сравнения ?:

Входные параметры

Номер параметра	Название	Тип	Допустимые значения
1	x	int	[-100, 100]
2	y	int	[-100, 100]

Выходные параметры

Вывести на экран целое значение функции.

Таблица тестирования

x	y	Результат
0	0	0
0	1	0
1	0	1
1	1	2
-1	0	-5
0	-1	0
-1	-1	1
1	-1	0
-1	1	-5
-25	85	-125
-41	36	-205
35	-52	-17
-77	85	-385
64	-27	37
42	83	125

Листинг

```
#include <stdio.h>

#include <test_utils.h>

int main(int argc, char* argv[])
{
  ARG(1, int, x);
  ARG(2, int, y);
   printf("%d", x > 0 ? x + y : (x <= 0 && y < 0 ? x * y : 5 * x)
    );
   return 0;
}
```

Задание 2. Функция

Вычислить значение функции:

$$z = x^2 - \cos \frac{\ln \sqrt{|x|}}{\operatorname{tg} e^{-x}}$$

Необходимо определить ООФ и добавить проверки на входные параметры. Для вычисления значений функций воспользуйтесь математической библиотекой math.h

Входные параметры

Номер параметра	Название	Тип	Допустимые значения
1	x	double	[-100, 100]

Выходные параметры

Вывести на экран вещественное значение функции с точностью до 2-х знаков. Если значение выходит за ООФ, выводить "Не определено".

Таблица тестирования

x	Результат
-1.0	0.00
1.0	0.00
0.0	-nan
0.00001	-0.29
-0.00001	-0.29
1.1111	0.25
1000	-nan
-1000	1000000.95
-88.605532	7851.56
-86.218128	7434.18
-17.706825	313.40
-3.214558	9.50
-15.820694	250.11

Листинг

```
#include <math.h>
#include <stdio.h>
#define E_NUM 2.71828

#include "test_utils.h"

int main(int argc, char* argv[])
{
  ARG(1, double, x);
   printf("%.2lf", pow(x, 2) - cos(log(sqrt(fabs(x))) / tanh(pow(
   E_NUM, -x)) ));
   return 0;
}
```

Задание 3. Координатная плоскость

Вывести на экран номер четверти, которой принадлежит точка с координатами (x,y), или указать, какой оси принадлежит эта точка. Необходимо использовать вложенный условный оператор.

Входные параметры

Номер параметра	Название	Тип	Допустимые значения
1	x	int	[-100, 100]
2	y	int	[-100, 100]

Выходные параметры

Вывести целое значение, обозначающее номер четверти: 1 – правая верхняя, 2 – левая верхняя, 3 – левая нижняя, 4 – правая нижняя. Если точка лежит на одной из координатных осей, то необходимо вывести 0.

Таблица тестирования

x	y	Результат
0	0	0
0	1	0
1	0	0
1	1	1
0	-1	0
-1	0	0
-1	-1	3
1	-1	0
-1	1	2
-44	31	2
-70	24	2
-95	90	2
-52	-26	3
-83	-14	3
-100	9	2

Листинг

```
#include <stdio.h>

#include "test_utils.h"

int main(int argc, char* argv[])
{
  ARG(1, int, x);
```

```
  ARG(2, int, y);
  int res = 0;
  if(y > 0)
  {
  if(x > 0)
  {
    res = 1;
  }
  else if(x < 0)
  {
    res = 2;
  }
  }
  else if(y < 0)
  {
  if(x < 0)
  {
    res = 3;
  }
  else if(x < 0)
  {
    res = 4;
  }
  }
  printf("%d", res);
  return 0;
}
```

Задание 4. Последовательность

На вход подаётся натуральное число N. Вывести на экран последовательность вида "S=2*4*6*...*N если N четное, или "S=1*3*5*...*N если N нечетное. Необходимо использовать цикл for.

Входные параметры

Номер параметра	Название	Тип	Допустимые значения
1	N	int	[2, 30]

Выходные параметры

Вывести на экран последовательность чисел. Например, при N = 7 вывод должен быть таким: "S=1*3*5*7".

Таблица тестирования

N	Результат
2	S=2
3	S=1*3
4	S=2*4
5	S=1*3*5
15	S=1*3*5*7*9*11*13*15
20	S=2*4*6*8*10*12*14*16*18*20
15	S=1*3*5*7*9*11*13*15
5	S=1*3*5
3	S=1*3
29	S=1*3*5*7*9*11*13*15*17*19*21*23*25*27*29
20	S=2*4*6*8*10*12*14*16*18*20

Листинг

```
#include <stdio.h>

#include "test_utils.h"

int main(int argc, char* argv[])
{
  ARG(1, int, N);
   printf("S=");
   for(int i = 2 - N % 2; i <= N; i+=2)
   {
   printf("%d", i);
   if(i != N)
     printf("*");
   }
   return 0;
}
```

Задание 5. Число Фиббоначи

Вывести первое найденное число Фибоначчи, большее заданного n. Каждый член последовательности Фибоначчи является суммой двух предыдущих: xn = xn-1 + xn-2, x0 = 0, x1 = 1. Необходимо использовать цикл while.

Входные параметры

Номер параметра	Название	Тип	Допустимые значения
1	n	int	[1, 1000]

Выходные параметры

Вывести на экран найденное число.

Таблица тестирования

n	Результат
1	2
5	8
1000	1597
31	34
192	233
138	144
910	987
211	233
133	144

Листинг

```
#include <stdio.h>

#include "test_utils.h"

int main(int argc, char* argv[])
{
  ARG(1, int, n);
  int a = 0, b = 1, temp;
  while(b <= n)
  {
  temp = b;
  b += a;
  a = temp;
  }
  printf("%d", b);
  return 0;
}
```

Задание 6. Сократить дробь

M и N – числитель и знаменатель обыкновенной дроби. Составить программу, позволяющую сократить эту дробь. Необходимо использовать цикл do ... while.

Входные параметры

Номер параметра	Название	Тип	Допустимые значения
1	M	int	[-100, 100]
2	N	int	[-100, 100]

Выходные параметры

Вывести сокращённую дробь на экран в формате "целая-часть числитель/знаменательпример: 18/8> 2 1/4".

Таблица тестирования

M	N	Результат
0	0	-
0	1	0
1	0	-
1	1	1
16	16	1
-16	-4	-4
-3	-66	-1/22
120	1	120
27	13	2 1/13
123	-30	-4 1/10
36	-59	-36/59
42	51	42/51
-34	-67	-34/67
-91	92	-91/92
96	-35	-2 26/35
-48	4	-12

Листинг

```
#include <stdio.h>

#include "test_utils.h"

int main(int argc, char* argv[])
{
  ARG(1, int, M);
  ARG(2, int, N);
   if(N == 0)
  {
     printf("-");
     return 0;
  }
```

```
  char is_negative = M < 0 || N < 0;
  M = M < 0 ? -M : M;
  N = N < 0 ? -N : N;

  int whole = M / N;
  M -= N * whole;
  while(M > 1 && N % M == 0)
  {
    N /= M;
    M /= M;
  }
  if(whole || !M)
  {
    if(is_negative)
      printf("-");
    printf("%d", whole);
  }
  if(M)
  {
    if(whole)
      printf(" ");
    if(is_negative && !whole)
      printf("-");
    printf("%d/%d", M, N);
  }
  return 0;
}
```

Задание 7. Сумма бесконечного ряда

Вывести на экран значение суммы бесконечного ряда

$$S = 1 - \frac{x}{1!} + \frac{x^2}{2!} + ... + (-1)^n * \frac{x^n}{n!}$$

С заданной точностью

$$E = 10^{-4}$$

Сравнить полученное значение с целевым можно через функцию:

$$f = e^{-}x$$

Вычисление необходимо производить с помощью цикла, используя значения с предыдущих итераций.

Входные параметры

Номер параметра	Название	Тип	Допустимые значения
1	x	double	[-10, 10]

Выходные параметры

Вывести на экран вещественное число с точностью до указанного знака.

Таблица тестирования

x	Результат
-1.0	2.7183
1.0	0.3679
0.0	1.0000
0.5	0.6065
-0.5	1.6487
5.0	0.0067
-4.5	90.0171
-3.437609	31.1125
3.689325	0.0250
5.660770	0.0035
8.896451	0.0001
-1.615731	5.0316

Листинг

```
#include <math.h>
#include <stdio.h>
#define E_NUM 2.71828

#include "test_utils.h"

int main(int argc, char* argv[])
{
  ARG(1, double, x);
  double sum = 1;
  double x_mul = x;
  int fact = 1;
  char sign = -1;
  double a = 1;
  while(fabs(a) > 0.0001)
  {
  a = sign * x_mul;
  for(int i = 1; i <= fact; i++)
```

```
    a /= i;
  sum += a;
  x_mul *= x;
  fact++;
  sign *= -1;
  }
  printf("%.4lf", sum);
  return 0;
}
```

Задание 8. Сумма бесконечного ряда

Вывести на экран значение суммы бесконечного ряда

$$f(x) = \frac{x * \cos\frac{\pi}{3}}{1} + \frac{x^2 * \cos\frac{2*\pi}{3}}{2} + ... + \frac{x^n * \cos n * \frac{\pi}{3}}{n} + ...$$

. С заданной точностью

$$E = 10^{-6}$$

. Значение функции для проверки:

$$y = -\frac{1}{2} * \ln 1 - 2 * x * \cos\frac{\pi}{3} + x^2$$

Вычисление необходимо производить с помощью цикла, используя значения с предыдущих итераций.

Входные параметры

Номер параметра	Название	Тип	Допустимые значения
1	x	double	[0.1, 0.8]

Выходные параметры

Вывести на экран вещественное число с точностью до указанного знака.

Таблица тестирования

x	Результат
0.1	0.047206 0.047206
0.8	0.087615 0.087616
0.100001	0.047206 0.047206
0.4	0.137460 0.137461
0.799999	0.087615 0.087616
0.491693	0.144096 0.144096
0.692686	0.120087 0.120086
0.456511	0.142861 0.142861
0.647146	0.129997 0.129995
0.198806	0.086858 0.086859

Листинг

```
#include <math.h>
#include <stdio.h>
#define PI_NUM 3.14

#include "test_utils.h"

int main(int argc, char* argv[])
{
  ARG(1, double, x);
  double sum = 0;
  double x_mul = x;
  int n = 1;
  double a;
  do
  {
  a = x_mul * cos(n * PI_NUM/3.) / n;
  sum += a;
  x_mul *= x;
  n++;
  }
  while(fabs(a) > 0.000001);
  printf("%lf ", -0.5 * log(1 - 2 * x * cos(PI_NUM/3) + x * x));
  printf("%.6lf", sum);
  return 0;
}
```

Задание 9. Не простые числа

Дано целое k. Вывести на экран все числа из диапазона [2,k], не являющиеся простыми. Простые числа не имеют других делителей, кроме 1 и самого себя. Необходимо использовать вложенные циклы.

Входные параметры

Номер параметра	Название	Тип	Допустимые значения
1	k	int	[2, 100]

Выходные параметры

Вывод чисел в формате: "2,4,6,8,".

Таблица тестирования

k	Результат
2	2,
5	2,3,5,
47	2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,
71	2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,
3	2,3,
48	2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,
4	2,3,
94	2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,
4	2,3,

Листинг

```
#include <math.h>
#include <stdio.h>

#include "test_utils.h"

int main(int argc, char* argv[])
{
  ARG(1, int, k);
  for(int i = 2; i <= k; i++)
  {
  int limit = sqrt(i);
  char is_prime = 1;
  for(int j = 2; j <= limit; j++)
  {
    if(i % j == 0)
    {
      is_prime = 0;
      break;
    }
  }
  if(is_prime)
    printf("%d,", i);
  }
  return 0;
}
```