**Правительство Российской Федерации**

**Федеральное государственное автономное образовательное учреждение высшего профессионального образования   
"Национальный исследовательский университет   
"Высшая школа экономики"**

Московский институт электроники и математики Национального

исследовательского университета "Высшая школа экономики"

Департамент прикладной математики

**ОТЧЕТ**

**По лабораторной работе №1**

***ВЫЧИСЛЕНИЕ МАТЕМАТИЧЕСКОГО ВЫРАЖЕНИЯ. ФОРМАТНЫЙ ВЫВОД***

**По курсу «Алгоритмизация и программирование»**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | ФИО студента | | Номер группы | Дата |
| Кармаев Александр Андреевич | БПМ211 | 20.09.2021 |
|  |
|  |
|  |

**Москва – 2021 г.**

**ЗАДАНИЕ (вариант №13)**

*Даны целые числа x, y и вещественное z. Вычислить a и b.*

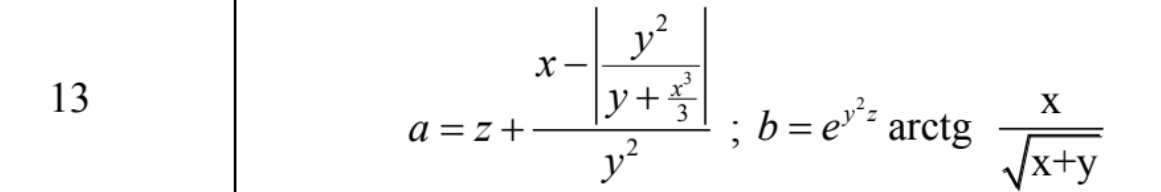
*Используя форматный ввод/вывод, организуйте дружественный интерфейс ввода данных*

*для решения задачи, а затем выведите на экран значения x, y и z (последнее в экспоненциальной форме с точностью 5 знаков после десятичного разделителя). Каждое значение выводить на новой строке, ширина поля – 10.*

*Вычисленные значения a и b выведите в десятичном формате с точностью 4 знака после*

*десятичного разделителя. Целое x отобразите в восьмеричном представлении, а y – в*

*шестнадцатеричном.*

****

**РЕШЕНИЕ**

#include <stdio.h> // #include – это директива для включения содержимого файла в программу // stdio.h – это cтандартная библиотеки ввода/вывода

#define \_USE\_MATH\_DEFINES // #define – это директива для создания макроса (программный объект) идентифицирующего токен (базовый программный объект) // \_USE\_MATH\_DEFINES – это директива препроцессора для использования математических констант из файла math.h

#include <math.h> // math.h – это библиотека, определяющая набор функций для выполнения общих математических операций и преобразований

int main() { // main – основная функция программы, с которой начинается выполнение программы и в которой всё обычно заканчивается

// инициализация переменных x, у и z для расчёта переменных а и b по формуле в условии

int x = 1, y = 2; // int – integer – спецификатор (ключевое слово) для целочисленного типа данных

long float z = 1.5, a = 2.5, b = 0.5; // float – тип данных чисел с плавающей точкой // long – модификатор необходимый для создания дополнительной памяти для хранения значения переменной

char answer[256]; // строка (массив символов типа char) для чтения значения введённого пользователем // 256 – размер массива, его длинна

// пользователь вводит значение для переменной x

printf("Enter integer number: x = "); // функция из библиотеки stdio.h, которая возвращает целое число (т.к. функция типа int). При успешном выполнении функции будет выведено число символов строки, при ошибки выведется отрицательное число

fgets(answer, sizeof(answer), stdin); // функция из библиотеки stdio.h, которая считывает определённое количество символов (2-ой параметр) из потока (3-ий параметр функции) и записывает значение в массив символов (строку) (1-ый парметр). В случае успеха функция возвращает указатель (переменная, содержащая адрес другой переменной) на строку, иначе 0 // sizeof() – оператор, возвращающий длину параметра в байтах // stdin – стандартный поток ввода

while (sscanf(answer, "%d", &x) != 1) { // используем цикл с предусловием, чтобы понять, когда пользователь введёт число, а не строку из буквенных символов // sscanf – функция идентичная scanf, лишь с тем различием, что она считывает данные из массива, первого параметра, и возвращает количество полей, значения которых были присвоены переменным или 0, если ни одно из полей не было присвоено переменным. 2-ой параметр sscanf – спецификатор ввода значения, 3-ий параметр – указывает с помощью амперсанта переменную, которой должно быть присвоено введённое значение

printf("Incorrect input. Try again: "); // выводим сообщение об ошибке

fgets(answer, sizeof(answer), stdin); // и заново считываем строку

}

// пользователь вводит значение для переменной y

printf("Enter integer number: y = ");

// данная конструкция while позволяет проверить введённое пользователем значение и не записать в переменную числового типа буквенные символы, что приведёт к ошибке и благодаря while пользователь будет вынужден вводить новые значения до тех пор, пока не введёт то, что нужно

fgets(answer, sizeof(answer), stdin);

while (sscanf(answer, "%d", &y) != 1) {

printf("Incorrect input. Try again: "); // функции передаётся только управляющая строка, без аргументов

fgets(answer, sizeof(answer), stdin);

}

// пользователь вводит значение для переменной z

printf("Enter float number: z = ");

fgets(answer, sizeof(answer), stdin);

while (sscanf(answer, "%Lf", &z) != 1) {

printf("Incorrect input. Try again: ");

fgets(answer, sizeof(answer), stdin);

}

// рассчитываем значение переменных а и b по формуле с помощью функций из библиотеки math.h

a = z + (x - fabs(pow(y, 2) / (y + pow(x, 3) / 3))) / pow(y, 2);

// fabs() – функция библиотеки math.h, возвращающая модуль значения параметра с плавающей точкой

// pow() - функция библиотеки math.h, возвращающая значения первого параметра функции, возведённого в степень, равную значению второго параметра

b = pow(M\_E, z \* pow(y, 2)) \* atan(x / sqrt((double)x + (double)y));

// M\_E – математическая константа равная экспоненте // exp(z \* pow(y, 2)) – другой способ получить значение экспоненты, возведённой в степень, указанную в параметре функции exp из библиотеки math.h

// atan() - функция библиотеки math.h, возвращающая арктангенс значения параметра в радианах

// sqrt() - функция библиотеки math.h, возвращающая корень значения параметра

// (double)ｘ – способ преобразовать переменную х к типу double

// следующий набор функций printf из библиотеки stdio.h позволяет нам вывести значения всех численных переменных в необходимом формате с помощью спецификаторов

printf("a = %10.4Lf\n", a); // printf – функция, записывающая в поток stdout значения аргументов (2, 3, … параметр функции) под управлением строки (1-ый параметр) в определённом формате, указанном с помощью спецификаторов (%(признаки)(ширина поля)(точность)(код формата)) и обычных символов. Количество и порядок спецификаторов соответствует аргументам.

printf("b = %10.4Lf\n", b);

printf("x = %10o\n", x);

printf("y = %10X\n", y);

printf("z = %10.5E\n", z);

return 0; // оператор int типа (т.к. функция main() имеет int тип данных), который завершает выполнение функции main() и возвращает значение типа integer, 0 – число сообщающее, что ошибок в программе не найдено, всё выполнено успешно // на самом деле данную строчку в функции main() можно и не писать, так как она сама сделает это неявно, если опустить данную строку

}

**ТЕСТЫ**

**Тест № 1**

Enter integer number: x = 1

Enter integer number: y = 1

Enter float number: z = 1.5

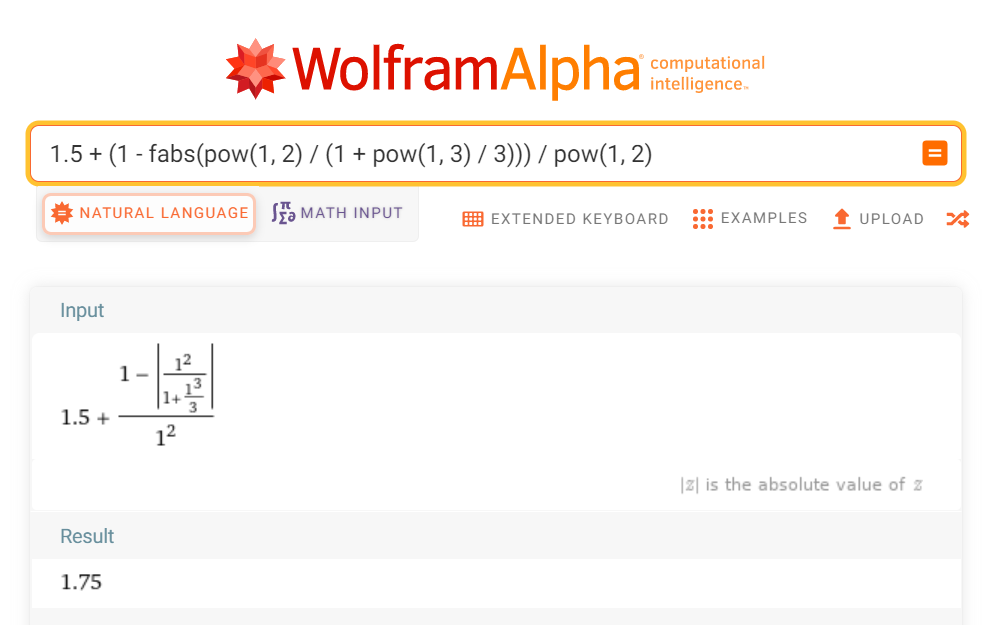
a = 1.7500

b = 2.7584

x = 1

y = 1

z = 1.50000E+00

Изображение выглядит как текст

Автоматически созданное описание

**Тест № 2**

Enter integer number: x = 23

Enter integer number: y = 9

Enter float number: z = 2.0

a = 2.2837

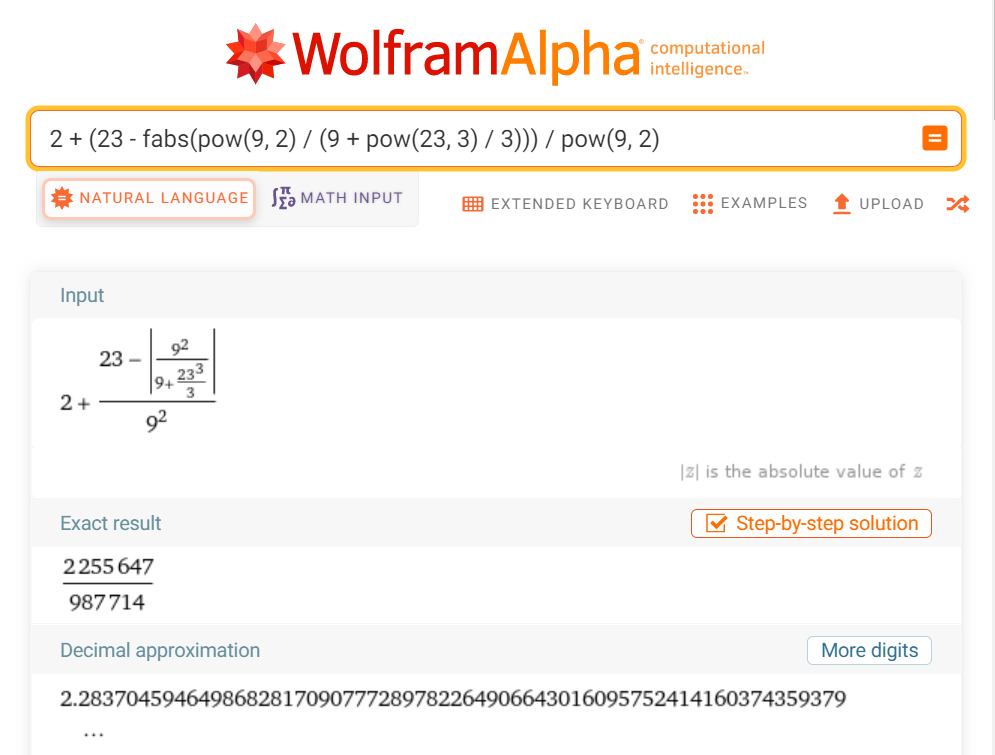
b = 30160449626417508789542247463017620553404675968077028275538911989596160.0000

x = 27

y = 9

z = 2.00000E+00

// число b вышло таким большим, потому что по формуле мы возводим число е в степень y^2\*z и при наших входных значениях степень равна 162, что очень немало

Изображение выглядит как текст

Автоматически созданное описание

**Тест № 3**

Enter integer number: x = 6463528

Enter integer number: y = 8383729

Enter float number: z = 16343.46483

a = 16343.4648

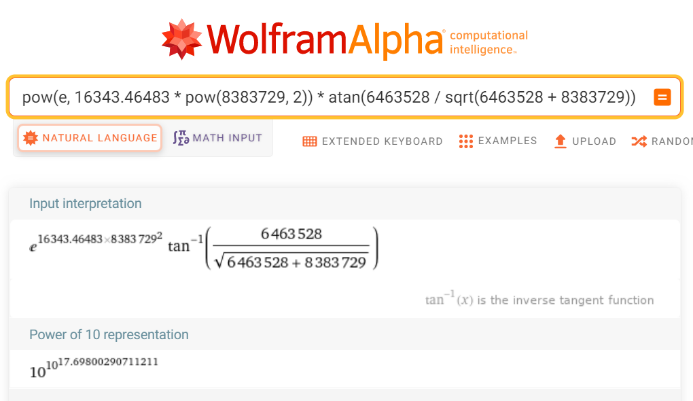
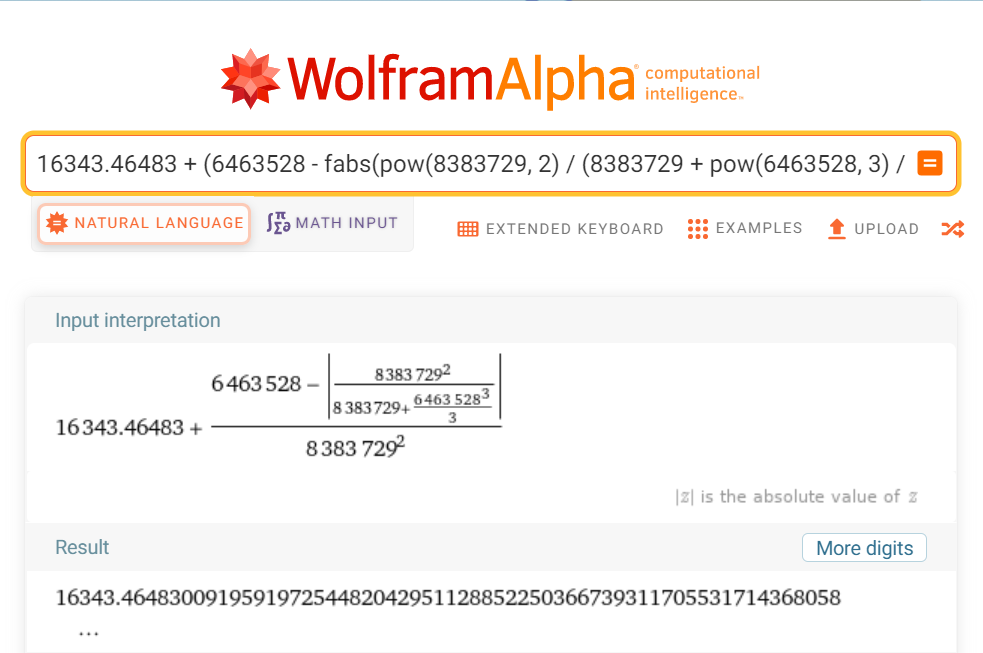
b = inf

x = 30520050

y = 7FECF1

z = 1.63435E+04

// данный пример является логичным продолжением 2-ого: мы взяли уже очень большие числа y и z, из-за чего получили переполнение числа b (inf – infinity – бесконечность)



**Тест № 4**

Enter integer number: x = -1

Enter integer number: y = 2

Enter float number: z = -1

a = -1.8500

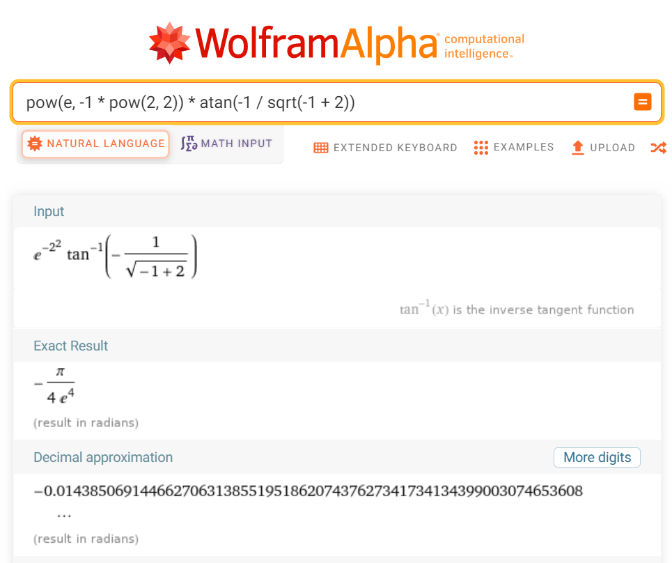
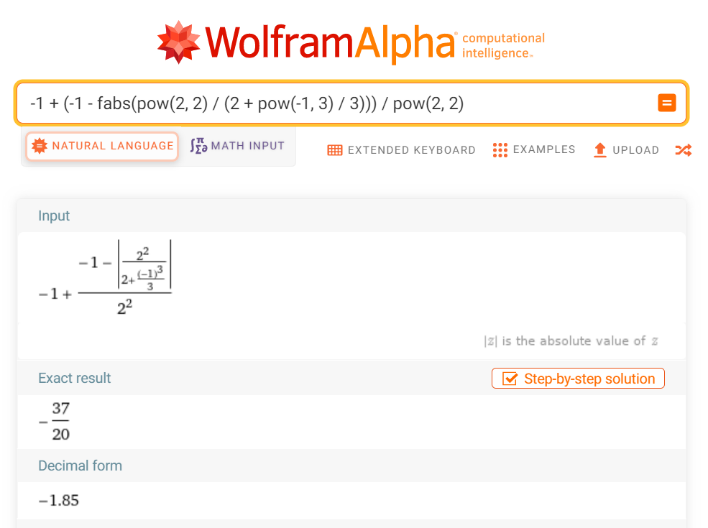
b = -0.0144

x = 37777777777

y = 2

z = -1.00000E+00

// В данном примере мы начали использовать отрицательные числа. Но теперь выражение e^(y^2\*z) будет наоборот становиться очень маленьким, из-за отрицательной степени и значение переменной b тоже. Также значение x было напечатано в 8-чной системе, и оно выглядит очень большим и не похоже на 1 и так же не имеет знака “-“, однако всё верно, так как программа вывела число с использованием дополнительного кода.



**Тест № 5**

Enter integer number: x = -5

Enter integer number: y = 6

Enter float number: z = -10

a = -10.1669

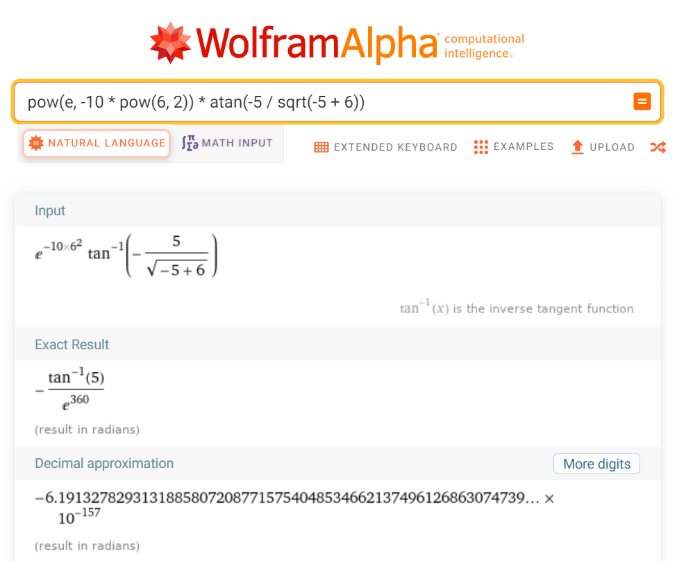
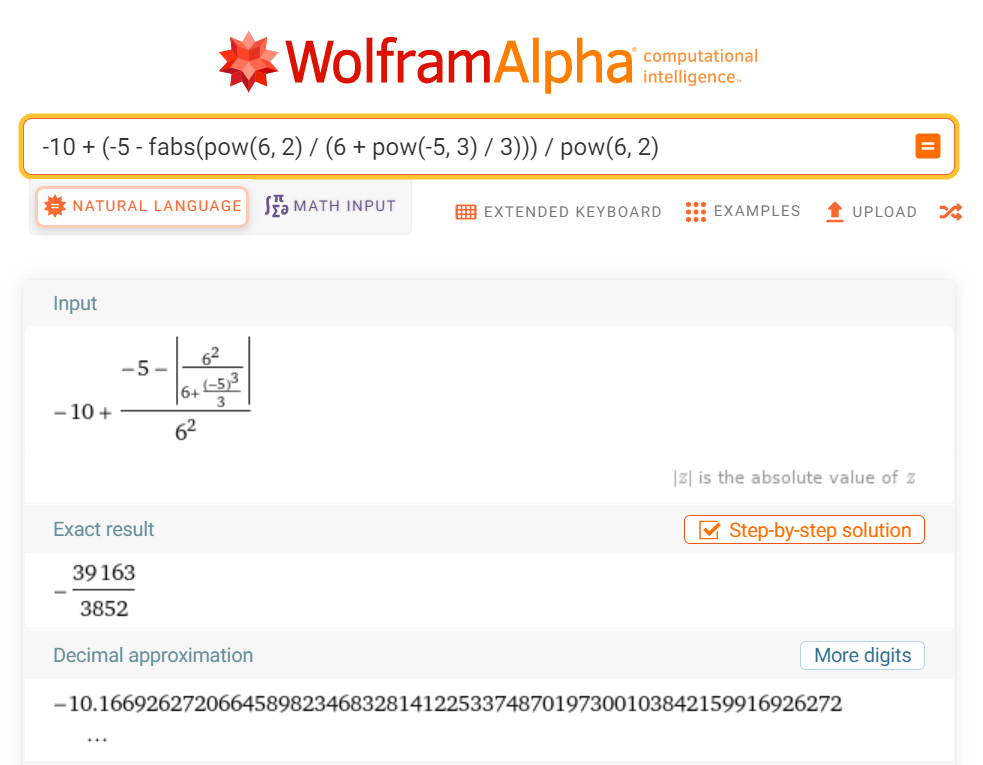
b = -0.0000

x = 37777777773

y = 6

z = -1.00000E+01

// теперь число b стало равно нулю, однако это не совсем верно: дело в том, что мы используем сокращение точности до 4 знаков после запятой, поэтому число b имеет такой вид, а на самом деле оно не равно нулю, оно просто очень маленькое, из-за возведения экспоненты в отрицательную степень



**Тест № 6**

Enter integer number: x = 1

Enter integer number: y = -1

Enter float number: z = 1

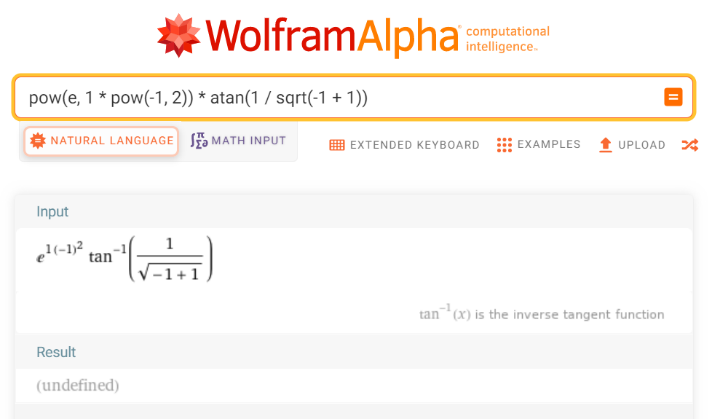
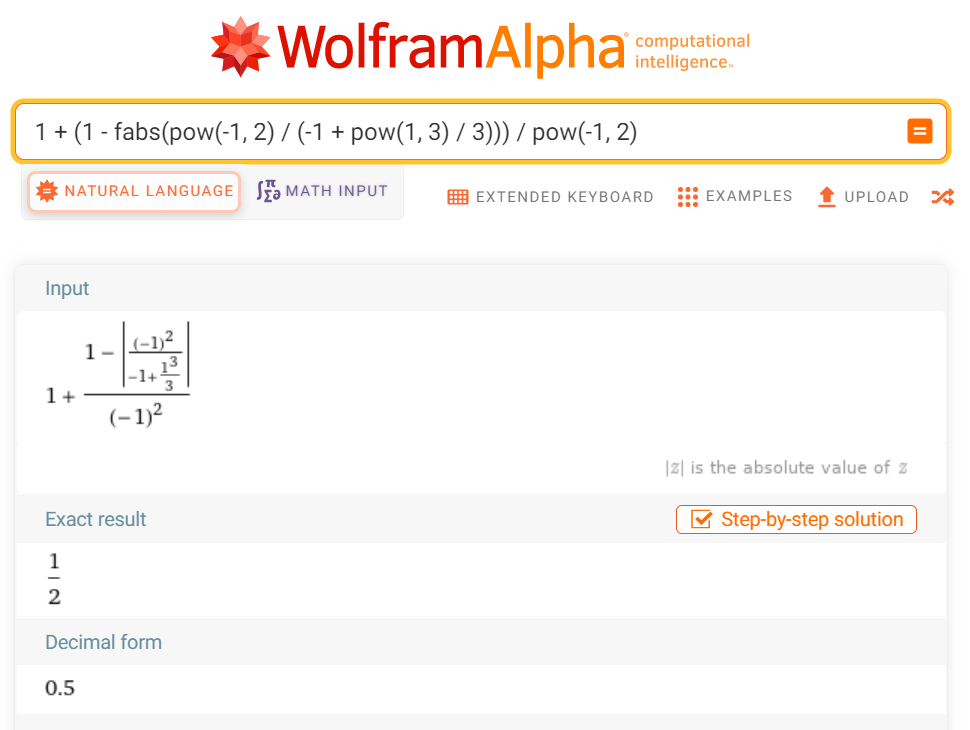
a = 0.5000

b = 4.2699

x = 1

y = FFFFFFFF

z = 1.00000E+00

// теперь по формуле знаменатель аргумента арктангенса равен 0, то есть аргумент равен бесконечности (inf), однако никакой ошибки в переменной b у нас нет, дело в том, что функция atan при аргументе равном inf вернёт значение 90 градусов в радианах (~ 1.57) 

**Тест № 7**

Enter integer number: x = -1

Enter integer number: y = -1

Enter float number: z = -1

a = -2.7500

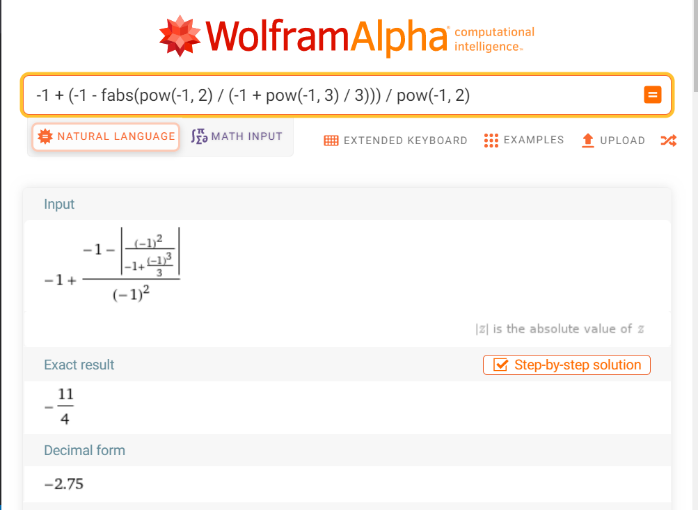
b = -nan(ind)

x = 37777777777

y = FFFFFFFF

z = -1.00000E+00

// теперь значение b равно -nan(ind) – ошибка неопределённости – сообщает о том, что мы производим математическую операцию, не имеющую смысла, в нашем случае это извлечение квадратного корня из отрицательного числа

Изображение выглядит как текст

Автоматически созданное описание

**Тест № 8**

Enter integer number: x = 0

Enter integer number: y = 0

Enter float number: z = 0

a = nan

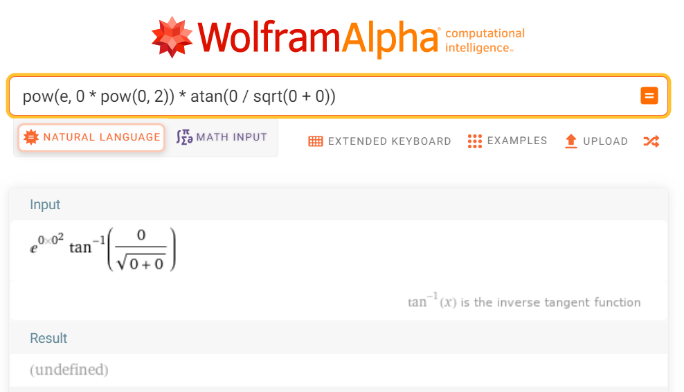
b = -nan(ind)

x = 0

y = 0

z = 0.00000E+00

// на этот раз переменная а тоже выдаёт ошибку nan – Not a Number, всё из-за того, у нас возникает многочисленное деление на ноль в формуле. Однако в переменной b на этот раз возникла ошибка -nan(ind), из-за деления нуля на ноль.



**Тест № 9**

Enter integer number: x = na

Incorrect input. Try again: 1

Enter integer number: y = hf

Incorrect input. Try again: hf

Incorrect input. Try again: 1

Enter float number: z = 1.2

a = 1.4500

b = 2.0435

x = 1

y = 1

z = 1.20000E+00

// данный пример демонстрирует работу моего обработчика входных значений

