Государственное образовательное учреждение высшего образования

«Уральский федеральный университет имени первого Президента России Б.Н. Ельцина»

090302 200008 006

**«Технологии обработки информации»**

Семестр 4

**ОТЧЁТ**

по лабораторным работам

Лабораторная работа №2



Преподаватель: Плотников В.Ю.

Студент : Киселев В.А.

Группа : ФТ — 200008

Екатеринбург 2022 г.

Оглавление

[Цель работы 3](#_Toc104577268)

[Постановка задачи 3](#_Toc104577269)

[Средства разработки 3](#_Toc104577270)

[Описание алгоритма 3](#_Toc104577271)

[Реализация 6](#_Toc104577272)

[С++ 6](#_Toc104577273)

[Python 6](#_Toc104577274)

[JavaScript 7](#_Toc104577275)

[Результаты 8](#_Toc104577276)

[C++ 8](#_Toc104577277)

[Python 8](#_Toc104577278)

[JavaScript 8](#_Toc104577279)

[Анализ результатов 8](#_Toc104577280)

[Выводы 8](#_Toc104577281)

# Цель работы

Экспериментально установить пределы представимых значений для вещественных типов данных в следующих языках программирования: C++ (для типов float, double и long double), Python и JavaScript. На основании полученных результатов сделать заключение о способе представления и размере. Оформить отчёт о проделанной работе.

# Постановка задачи

1) Разработать алгоритм

2) Реализация алгоритма

3) Провести эксперимент

4) Анализ результатов

5) Выводы

# Средства разработки

1) CLion 2021.3.3

2) PyCharm Professional 2021.3.2

3) WebStorm 2021.3.2

4) Google Chrome 98.0.4758.102

# Описание алгоритма

Первый вопрос, на который необходимо ответить – как определить, что именно данное значение максимально. В отличие от целых типов тут может записаться не только нормальное число, но и бесконечность.

Максимальное число меньше бесконечности. При прибавлении любого числа к бесконечности получится бесконечность. Таким образом надо хранить три числа: число предыдущего шага, число текущего, число следующего. Если число текущего и число следующего равны, то они оба бесконечность, те больше, чем нам нужно. Таким образом можно определить выход за границу.

Шаблонная функция содержит 2 цикла while, один логарифмическим поиском доходит до момента, когда текущий шаг бесконечен. После чего из него выходим.

Второй уменьшает шаг, пока мы не упрёмся в шаг равный нулю. После этого прибавляет уже минимальный ненулевой шаг к числу, пока число не начнёт выпадать в бесконечность. Число за шаг до выпадения – искомое число.

Вторая шаблонная функция ищет минимальное положительное (больше нуля) число. В ней используется только один цикл while. Выход из него осуществляется в случае, когда шаг столь мал, что округлился до нуля.

Пока шаг можно уменьшить, мы уменьшаем его вдвое.

# Реализация

## С++

#include <iostream>  
#include <iomanip>  
  
using namespace std;  
  
template<typename T> T search\_max\_value(T base) {  
  
 T step = 1;  
 T temp1 = 0;  
 T temp2 = 1;  
  
 while (temp1 != temp2)  
 {  
 base = temp1;  
 temp1 = temp2;  
 temp2 += step;  
 step \*= 1.3;  
 }  
  
 while (base + step != base)  
 {  
 if (base + step == temp1)  
 {  
 step \*= 0.5;  
 }  
 else {  
 base += step;  
 }  
 }  
  
 return base;  
}  
  
  
template<typename T> T search\_min\_value(T base) {  
 T step = 1;  
  
 while (base - step != base)  
 {  
 if (base - step <= 0)  
 {  
 step \*= 0.5;  
 }  
 else {  
 base -= step;  
 }  
 }  
  
 return base;  
}  
  
void output(auto & var){  
 if ((std::string)typeid(var).name() == (std::string)"char"){  
 std::cout << typeid(var).name() << setprecision(16) << " min:" << (int)search\_min\_value(var) << ", max:" << (int)search\_max\_value(var) << std::endl;  
 } else {  
 std::cout << typeid(var).name() << setprecision(16) << " min:" << search\_min\_value(var) << ", max:" << search\_max\_value(var) << std::endl;  
 }  
}  
  
int main() {  
 float var1 = 1;  
 double var2 = 1;  
 long double var3 = 1;  
  
 output(var1);  
 output(var2);  
 output(var3);  
  
 return 0;  
}

## Python

def search\_a\_maximum\_number(base):  
 step = 1  
 temp1 = 0  
 temp2 = 1  
 counter = 0  
  
 while temp1 != temp2:  
  
 base = temp1  
 temp1 = temp2  
 temp2 = temp2 + step  
 step \*= 1.3  
  
 if counter > 2700:  
 return "Value boundary not found in 2700 steps"  
 else:  
 counter += 1  
  
 while base + step != base:  
  
 if base + step == temp1:  
 step \*= 0, 5  
 else:  
 base += step  
 return base  
  
  
def search\_a\_minimum\_number(base):  
  
 step = 1  
 counter = 0  
  
 while base - step != base:  
  
 if not base - step > 0:  
 step \*= 0.5  
 else:  
 base = base - step  
  
 if counter > 2700:  
 return "Value boundary not found in 2700 steps"  
 else:  
 counter += 1  
  
 return base  
  
  
base = 1  
  
print("Maximum: ", search\_a\_maximum\_number(base))  
  
print("Minimum: ", search\_a\_minimum\_number(base))

## JavaScript

function search\_a\_max\_number(base){  
  
 let temp1 = 0.0;  
 let temp2 = 1.0;  
 let step = 1.0;  
 let count = 0.0;  
  
 while (temp1 !== temp2) {  
  
 base = temp1;  
 temp1 = temp2;  
 temp2 = temp2 + step;  
 step \*= 1.3;  
  
 if (count > 2700) {  
 return "Value boundary not found in 2700 steps";  
 } else {  
 count += 1;  
 }  
  
 }  
  
 while (base + step !== base) {  
  
 if (base + step === temp1){  
 step \*= 0.5;  
 } else {  
 base += step ;  
 }  
  
 return base;  
  
 }  
  
}  
  
function search\_a\_min\_number(base){  
  
 let step = 1.0  
 let count = 0.0  
  
 while (base - step !== base){  
  
 if (base - step <= 0){  
 step \*= 0.5;  
 } else {  
 base = base - step;  
 }  
  
 if (count > 2700){  
 return "Value boundary not found in 2700 steps";  
 } else {  
 count += 1;  
 }  
  
 }  
  
 return base;  
  
}  
  
x = 1.0;  
  
***console***.log("Maximum: " + search\_a\_max\_number(x));  
  
***console***.log("Minimum: " + search\_a\_min\_number(x));

# Результаты

## C++

Изображение выглядит как текст

Автоматически созданное описание

## Python



## JavaScript



# Анализ результатов

Полученный при помощи программы на C++ результат совпадает со справочным материалом, что указывает, что алгоритм работает верно.

В случае использования языков, таких как: JavaScript и Python, память выделяется динамически, потому пройдя 2700 итераций поиска (т.е. дойдя до числа 4.43 × 10^307 мы не вышли за пределы в поиске максимального значения).

# Выводы

Цель работы была выполнена в полном объёме. C++ выделяет для фиксированное количество байт на хранения каждого типа данных. Максимальное значение Double и Long Double совпадает, так как используется 64 битная система. В других языках представление динамическое, то есть не имеет ограничений (не считая аппаратных, связанных с конечностью оперативной памяти).