KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

Programavimo kalbų teorija (P175B124)

Laboratorinių darbų ataskaita

Atliko:

IFF-1/9 gr. studentas

Nedas Liaudanskis

2023 m. vasario 14 d.

Priėmė:

Lekt. Guogis Evaldas

KAUNAS 2023

TURINYS

[C++ arba Ruby(L1) 3](#_Toc127294835)

[1.1. Darbo užduotis 3](#_Toc127294836)

[1.2. Programos tekstas: 4](#_Toc127294837)

[1.3. Pradiniai duomenys ir rezultatai: 8](#_Toc127294838)

# C++ arba Ruby(L1)

## Darbo užduotis

619 – Numerically Speaking

Paveikslėlis, kuriame yra žinutė

Automatiškai sugeneruotas aprašymas

Užduoties pilnai įvykdyti nepavyko, nes nesugebėjau perskaityti kintamųjų arba su jais atlikti skaičiavimų, kurie buvo didesni nei unsigned long long int tipo kintamasis. Todėl skaitydamas failus uždėjau skaičių limitą ir raidžių limitą (19 skaičių ir 13 raidžių). Tikiuosi tai nėra labai didelė problema.

## Programos tekstas:

// Lab1.cpp : This file contains the 'main' function. Program execution begins and ends there.

//

#include <iostream>

#include <fstream>

#include <list>

#include <iterator>

#include <array>

#include <iterator>

#include <string>

#include <vector>

#include <chrono>

#include <sstream>

using namespace std;

/// <summary>

/// An object Class that defines the word, its numerical meaning and holds the output string.

/// </summary>

class NumericalData

{

private:

/// <summary>

/// String type variable that has the word

/// </summary>

string Word;

/// <summary>

/// Int type variable that holds the numarical form of the word

/// </summary>

unsigned long long int Number;

/// <summary>

/// Output string

/// </summary>

string OutPut;

/// <summary>

/// Function that Converts the given word into numbers using the method given in the exercise

void StringToNumbers()

{

unsigned long long int Numb = 0;

unsigned long long int x = 1;

for (int i = Word.length() - 1; i >= 0; i--)

{

Numb += (Word[i] - 'a' + 1) \* x;

x = x \* 26;

}

this->Number = Numb;

}

/// <summary>

/// Function that converts numbers(int type variables) into string type words, using the method given in the exercise

/// </summary>

void NumbersToString()

{

string line = "";

unsigned long long int Saved = Number;

while (Number != 0)

{

line += (char)('a' + (Number - 1) % 26);

this->Number = (Number - 1) / 26;

}

reverse(line.begin(), line.end());

this->Word = line;

this->Number = Saved;

}

/// <summary>

/// Converts an integer into the required output string

/// </summary>

void ForOutPut()

{

string Out = to\_string(Number);

int count = 0;

for (int i = Out.length() - 1; i >= 0; i--)

{

count++;

if (count % 3 == 0 && i != 0)

{

Out.insert(i, ",");

}

}

this->OutPut = Out;

}

public:

/// <summary>

/// Constructor used for receiving only the number ar the given option

/// </summary>

/// <param name="Num"> long long int type value that show the number that will be neede to decrypt </param>

NumericalData(unsigned long long int Num)

{

Word = "";

Number = Num;

OutPut = "";

}

/// <summary>

/// Constructor used for receiving string type variable as the given option (the given word)

/// </summary>

/// <param name="word"> The string object that is the word that will be turned into numerical values </param>

NumericalData(string word)

{

Word = word;

Number = 0;

OutPut = "";

}

/// <summary>

/// The method used to calculate the remaining values of all the words and numerical numbers that are in this object

/// </summary>

void CalculateValues()

{

if (Word == "")

{

NumbersToString();

ForOutPut();

}

else

{

StringToNumbers();

ForOutPut();

}

}

/// <summary>

/// Returns the starting string (used for output)

/// </summary>

/// <returns> starting string(word) </returns>

string ReturnStartingString()

{

return Word;

}

/// <summary>

/// Returns the output string

/// </summary>

/// <returns> The output string </returns>

string ReturnOutput()

{

return OutPut;

}

};

/// <summary>

/// A not finished class that was used to test a way to store bigger integer numbers

/// </summary>

class Utils

{

int\* GetBiggerInteger(string str)

{

int x = str.size();

int\* Integer = new int[str.size()];

int a = 0;

while (a != x)

{

Integer[a] = str[a] - '0';

a++;

}

return Integer;

}

};

/// <summary>

/// A cointainer that holds the NumericalData objects

/// </summary>

class NumericalDataContainer

{

private:

/// <summary>

/// NumericalData object vector

/// </summary>

vector<NumericalData> AllData;

public:

/// <summary>

/// Constructor used in this class, creates an empty vector

/// </summary>

NumericalDataContainer()

{

AllData = {};

}

/// <summary>

/// A method used to add data to the container

/// </summary>

/// <param name="data"> NumericalData object tjat needs t be added to the container </param>

void AddData(NumericalData data)

{

AllData.push\_back(data);

}

/// <summary>

/// Return the Numericaldata object that was stored in this containers index

/// </summary>

/// <param name="index"> Index of the object we want to get </param>

/// <returns> NumericalData object that matches the given index </returns>

NumericalData ReturnDataByIndex(int index)

{

return AllData.at(index);

}

/// <summary>

/// Returns how many objects are stored inside the container

/// </summary>

/// <returns> How many objects are stored inside the container </returns>

int Count()

{

return AllData.size();

}

/// <summary>

/// A method used to calculate all NumericalData's missing values(results)

/// </summary>

void CalculateData()

{

int count = 0;

for (NumericalData Data : AllData)

{

Data.CalculateValues();

AllData.at(count) = Data;

count++;

}

}

};

/// <summary>

/// A class used for reading data from the ttxt file and printing results into the console

/// </summary>

class InOut {

public:

/// <summary>

/// Constructor

/// </summary>

InOut() {

};

/// <summary>

/// A static method that returns a container full of NumericalData objects, that are read from the txt file

/// </summary>

/// <param name="File"> Name of the file where the data is beeing stored at </param>

/// <returns> NumericalDataContainer object that is a container for all NumericalData objects in this project </returns>

static NumericalDataContainer ReadFile(string File)

{

NumericalDataContainer AllData;

fstream myFile;

myFile.open(File);

if (myFile.is\_open())

{

string line;

while (getline(myFile, line))

{

if (isdigit(line[1]))

{

if (line.length() <= 19)

{

unsigned long long int digit = stoll(line);

NumericalData Data(digit);

AllData.AddData(Data);

}

else

{

cout << "Number: " << line << " Was longer then the max length of 19 digits, so the calculations couldn't be done." << endl;

}

}

else

{

if (line.length() <= 13)

{

NumericalData Data(line);

AllData.AddData(Data);

}

else

{

cout << "Line: " << line << " Was longer then the max length of a string type objects (13 letters), so calculations couldn't be done." << endl;

}

}

}

}

myFile.close();

return AllData;

}

/// <summary>

/// Print the fial answers into the console window

/// </summary>

/// <param name="AllData"> Container storing all the data </param>

static void PrintOutPut(NumericalDataContainer AllData)

{

for (int i = 0; AllData.Count() > i; i++)

{

cout << "String: " << AllData.ReturnDataByIndex(i).ReturnStartingString() << " Number: " << AllData.ReturnDataByIndex(i).ReturnOutput() << endl;

}

}

};

/// <summary>

/// Main function used to run the program

/// </summary>

/// <returns></returns>

int main()

{

/// <summary>

/// std::chrono function used to get the time before the program runs,

/// that will be used to calculate the time it took for the program to do all the calculations.

/// </summary>

/// <returns></returns>

auto start = std::chrono::high\_resolution\_clock::now();

InOut Out;

NumericalDataContainer Data = Out.ReadFile("Duom.txt");

Data.CalculateData();

Out.PrintOutPut(Data);

auto stop = std::chrono::high\_resolution\_clock::now();

auto time = std::chrono::duration\_cast<std::chrono::seconds>(stop - start);

cout << "Time taken: " << time.count();

}

// Run program: Ctrl + F5 or Debug > Start Without Debugging menu

// Debug program: F5 or Debug > Start Debugging menu

// Tips for Getting Started:

// 1. Use the Solution Explorer window to add/manage files

// 2. Use the Team Explorer window to connect to source control

// 3. Use the Output window to see build output and other messages

// 4. Use the Error List window to view errors

// 5. Go to Project > Add New Item to create new code files, or Project > Add Existing Item to add existing code files to the project

// 6. In the future, to open this project again, go to File > Open > Project and select the .sln file

## Pradiniai duomenys ir rezultatai:

**Duomenys:**

|  |
| --- |
| 29697684282993  snowfall  28011622636823854456520  finally  zzzzzzzzzzzz  99246114928149462 |

**Rezultatai:**

|  |
| --- |
| Number: 28011622636823854456520 Was longer then the max length of 19 digits, so the calculations couldn't be done.  String: elementary Number: 29,697,684,282,993  String: snowfall Number: 157,118,051,752  String: finally Number: 1,966,850,729  String: zzzzzzzzzzzz Number: 99,246,114,928,149,462  String: zzzzzzzzzzzz Number: 99,246,114,928,149,462  Time taken: 0 |