KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

Programavimo kalbų teorija (P175B124)

Laboratorinių darbų ataskaita

Atliko:

IFF-1/9 gr. studentas

Martynas Kuliešius

2023 m. vasario 15 d.

Priėmė:

Lekt. Evaldas Guogis

Lekt. Tautvydas Fyleris

KAUNAS 2023

TURINYS

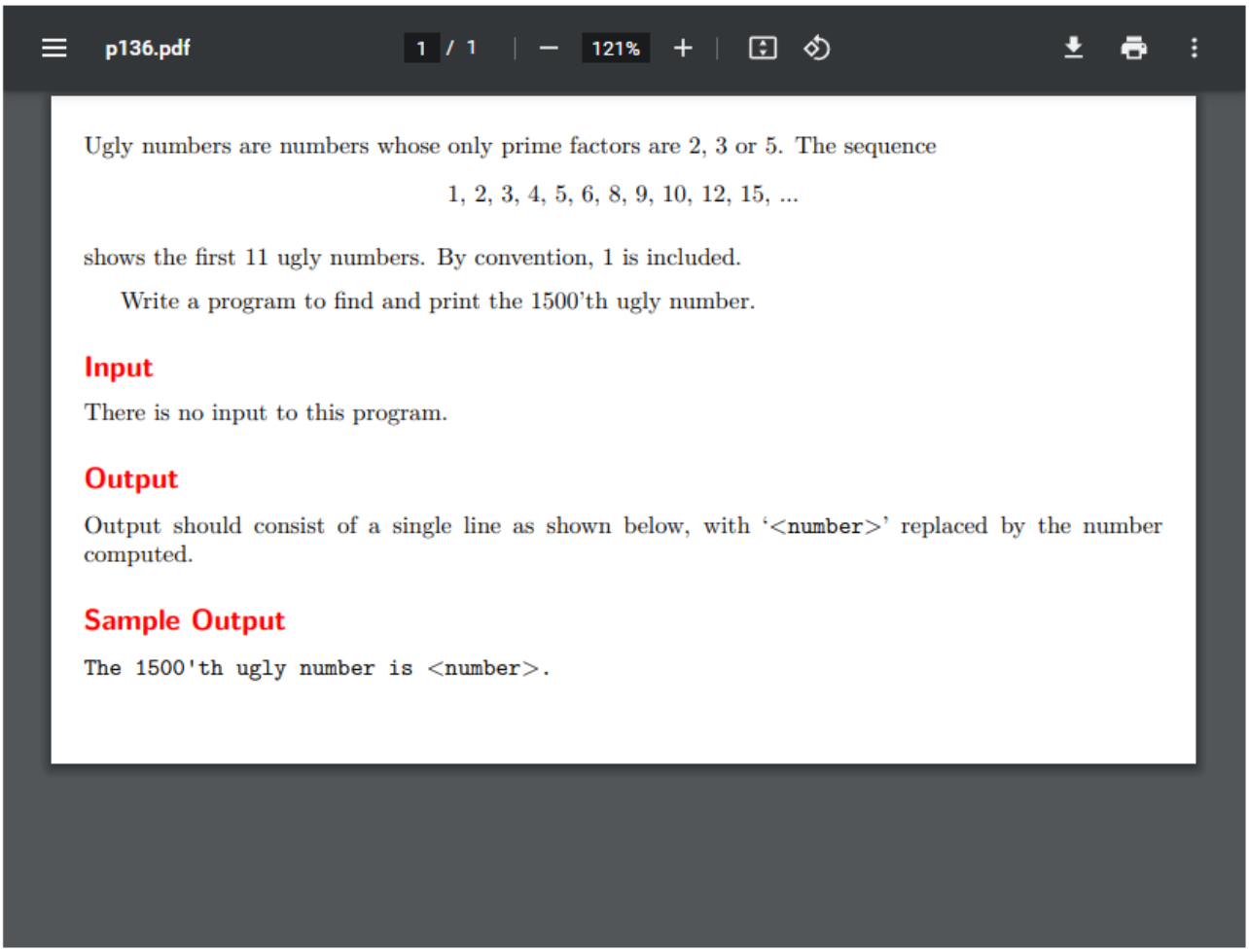
1. C++ arba Ruby (L1)

1.1. Darbo Užduotis

136 - Ugly Numbers

Debug PDF

Time limit: 3.000 seconds



Ugly numbers are numbers whose only prime factors are 2, 3 or 5. The sequence

1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, ...

shows the first 11 ugly numbers. By convention, 1 is included.

Write a program to find and print the 1500'th ugly number.

Input

There is no input to this program.

Output

Output should consist of a single line as shown below, with '<number>' replaced by the number computed.

Sample Output

```
The 1500'th ugly number is <number>.
```

1.2. Programos Tekstas

Main.cpp failas:

```
// 136 - Ugly Numbers
// Martynas Kuliešius IFF-1/9 E**36

/*
Užduoties sąlyga:

Ugly numbers are numbers whose only prime factors are 2, 3 or 5. The sequence
1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, ...
shows the first 11 ugly numbers. By convention, 1 is included.
Write a program to find and print the 1500’th ugly number.
--Input--
There is no input to this program.
--Output--
Output should consist of a single line as shown below, with ‘<number>’ replaced by
the number
computed.
--Sample Output--
The 1500'th ugly number is <number>.

*/


#include <iostream>
#include <vector>
#include <chrono>

using namespace std;

/// <summary>
/// Class to declare methods and write their headers
/// </summary>
class Utils {
public:
    /// <summary>
    /// Gets the minimum value out of three integer values
    /// </summary>
    /// <param name="a">first integer value</param>
    /// <param name="b">second integer value</param>
    /// <param name="c">third integer value</param>
    /// <returns>Returns minimum value of of three integer values</returns>
    int getMin(int a, int b, int c);

    /// <summary>
    /// Returns the n-th ugly number
    /// </summary>
    /// <param name="n"> index/number of the ugly number wanted</param>
    /// <returns> the wanted ugly number</returns>
    int returnUglyNumber(int n);

};

class InOut {
public:
    /// <summary>
    /// Header of n input method
    /// </summary>
    /// <returns> the integer value for n </returns>
    static int ReadN();
    /// <summary>
    /// header of method to print to screen.
    /// </summary>
    /// <param name="n"> the integer value for n </param>
```

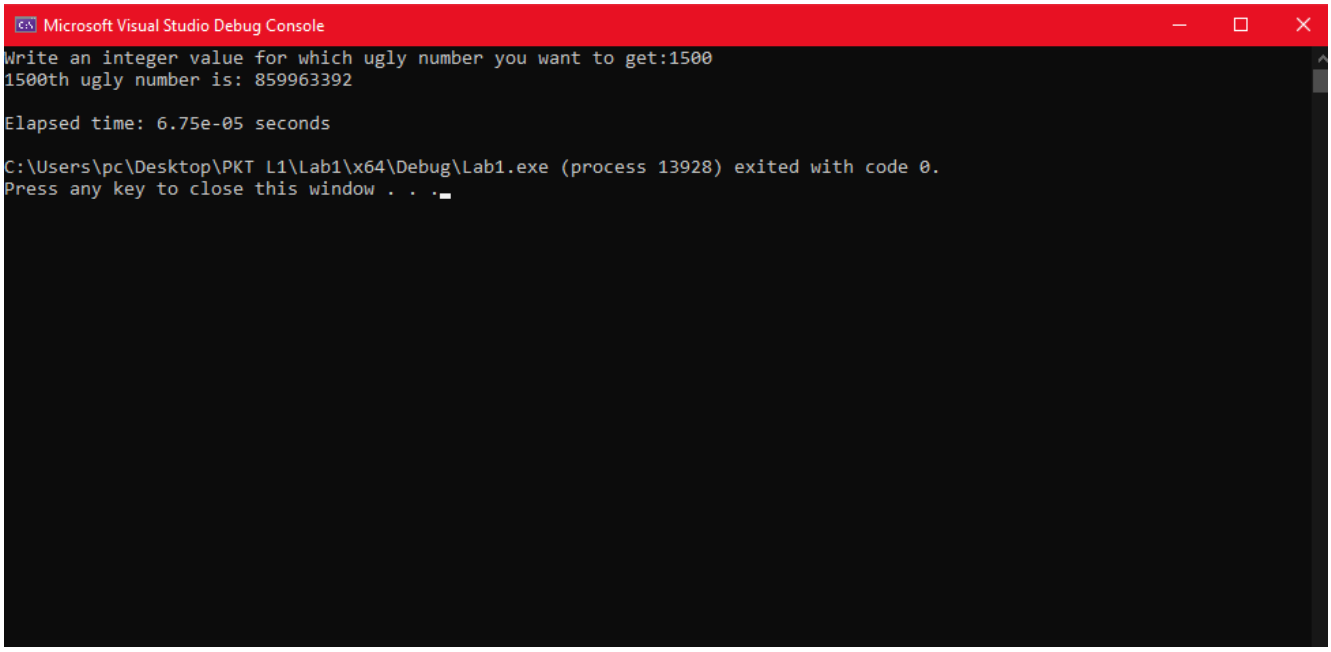
```
    /// <param name="uglyNumber"> the integer value for n-th ugly number </param>
    static void PrintToScreen(int n, int uglyNumber);
};



int main() {
    Utils workMethods; // create/declare class
    InOut inOut; // create/declare class
    chrono::time_point<std::chrono::system_clock> s, e; // start/end

    int n = inOut.ReadN(); // the nth number we want

    s = chrono::system_clock::now();
    int uglyNumber = workMethods.returnUglyNumber(n); // n-th ugy numebr we want
    e = chrono::system_clock::now();

    inOut.PrintToScreen(n, uglyNumber); //output ugly number to screen

    chrono::duration<double> time = e - s;
    cout << endl << "Elapsed time: " << time.count() << " seconds" << endl;

    return 0;
}

int Utils::getMin(int a, int b, int c) {
    return min(min(a, b), c);
}

int Utils::returnUglyNumber(int n) {
    vector<int> uglyNumbers(n); // array to store numbers
    uglyNumbers[0] = 1;
    //
    int i2 = 0;
    int i3 = 0;
    int i5 = 0;
    int nextMultipleOf2 = 2;
    int nextMultipleOf3 = 3;
    int nextMultipleOf5 = 5;


    // for loop loops util i index reaches the value of n, thus creating n ugly
numbers.
    for (int i = 1; i < n; i++) {
        int nextNumber = getMin(nextMultipleOf2, nextMultipleOf3,
nextMultipleOf5);
        uglyNumbers[i] = nextNumber;

        if (nextNumber == nextMultipleOf2) {
            i2++;
            nextMultipleOf2 = uglyNumbers[i2] * 2;
        }

        if (nextNumber == nextMultipleOf3) {
            i3++;
            nextMultipleOf3 = uglyNumbers[i3] * 3;
        }

        if (nextNumber == nextMultipleOf5) {
            i5++;
            nextMultipleOf5 = uglyNumbers[i5] * 5;
        }
    }
    return uglyNumbers[n - 1];
}
```

```
/// <summary>
/// Simple method to return n value
/// </summary>
/// <returns> the integer value of n </returns>
int InOut::ReadN() {
    int n;
    cout << "Write an integer value for which ugly number you want to get:";
    cin >> n;
    return n;
}
/// <summary>
/// Simple method to return the ugly number value
/// </summary>
/// <param name="n"> integer value of n </param>
/// <param name="uglyNumber"> integer value of the ugly number </param>
void InOut::PrintToScreen(int n, int uglyNumber) {
    cout << n << "th ugly number is: " << uglyNumber << endl;
}
```

1.3. Pradiniai Duomenys ir Rezultatai

Testas 1:

Pirmajam testui nusprendžiau įvesti užduotyje nurodytą įvesties reikšmę: 1500. Programa greitai suveikia ir per 0.0675 milisekundes duoda atsakymą: 859963392



Testas 2:

Antrajam testui nutariau pasirinkti vieną iš duotų reikšmių: 10. Programa greitai suveikia ir per 0.054 milisekundes duotą atsakymą: 12