# ЛАБОРАТОРНАЯ РАБОТА №2

# ЭКСПЕРИМЕНТАЛЬНАЯ ОЦЕНКА СРЕДНЕГО КОЛИЧЕСТВА ОПЕРАЦИЙ ПЕРЕПРИСВАИВАНИЯ В АЛГОРИТМЕ ПОИСКА МИНИМУМА

Цель работы

В ходе ряда экспериментов исследовать соответствие теоретической оценки трудоёмкости алгоритма поиска минимума, основанной на вычислении гармонического среднего. Получить практические навыки применения генераторов случайных чисел при проведении численных экспериментов.

Задания

1. Изучить функции, использующиеся для генерации псевдослучайных чисел (rand, random, randomize и т. п.), и привести их описание;
2. Написать программу поиска минимума в массиве сгенерированных псевдослучайных чисел и подсчёта n-го гармонического числа;
3. Подсчитать количество операций переприсваивания для программной реализации поиска минимума в массиве случайных чисел. Длину массива и максимальное случайное число в последовательности взять в соответствии с вариантом задания (Таблица 1);

|  |  |  |
| --- | --- | --- |
| Вариант | Наибольшее случайное число в последовательности | Количество элементов в массиве псевдослучайных чисел |
| 12 | 400 | 100, 200, 500 |

Таблица 1 – Вариант задания

1. Сравнить практически полученное значение с теоретическим n-м гармоническим числом. Сделать выводы по работе;

Текст программы

#include <iostream>

#include <cmath>

#include <ctime>

//#include <cstdlib>

using namespace std;

const int MAX = 401; // макс. значение числа в массиве

// Cоздание массива из N псевдослучайных целых чисел от 0 до MAX

void create\_array(int N, int arr[])

{

srand(time(NULL));

//randomize();

cout << "\nMax. integer - " << MAX - 1 << endl;

for (int i = 0; i < N; i++)

{

arr[i] = rand() % MAX;

cout << arr[i] << " ";

}

cout << endl << endl;

}

// Подсчёт количества операций переприсваивания в алгоритме поиска

// наим. числа массива и сравнение его с математическим ожиданием

int main()

{

int min; // значение минимума

int cnt; // счетчик операций переприсваивания

int N; // длина массива

cout << "Input amount of numbers - ";

cin >> N;

int vector[N]; // массив чисел

// подсчёт n-го гармонического числа

double Hn = log(N) + 0.57;

cout << "Harmonic number - " << Hn << endl;

// алгоритм поиска минимального числа

create\_array(N, vector);

min = vector[0];

cnt = 1;

for (int i = 1; i < N; i++)

if (vector[i] < min)

{

min = vector[i];

cnt++;

}

cout << "Min - " << min << endl;

cout << "Cout - " << cnt;

return 1;

}

Ход работы

Были изучены функции стандартной библиотеки языка C++, использующиеся для генерации псевдослучайных чисел:

**rand()** − возвращает псевдослучайное целое число в заданном диапазоне (по умолчанию от 0 до RAND\_MAX). При последующих запусках программы всего лишь отображает числа, сгенерированные в первый раз.

**random()** – возвращает случайное число в интервале от 0 до num-1.

**randomize()** – инициализирует генератор случайных чисел некоторым случайным числом. Использует функцию **time**.

**srand()**− устанавливает начальную точку, из которой происходит генерирование случайных чисел.

Написанная программа была запущена для указанных в варианте значений длины массива: 100, 200, 500.

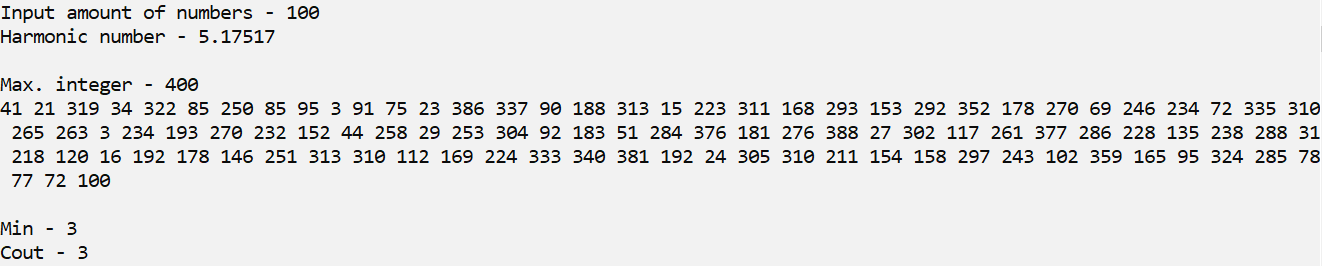


Рисунок 1 – Вычисления для N=100

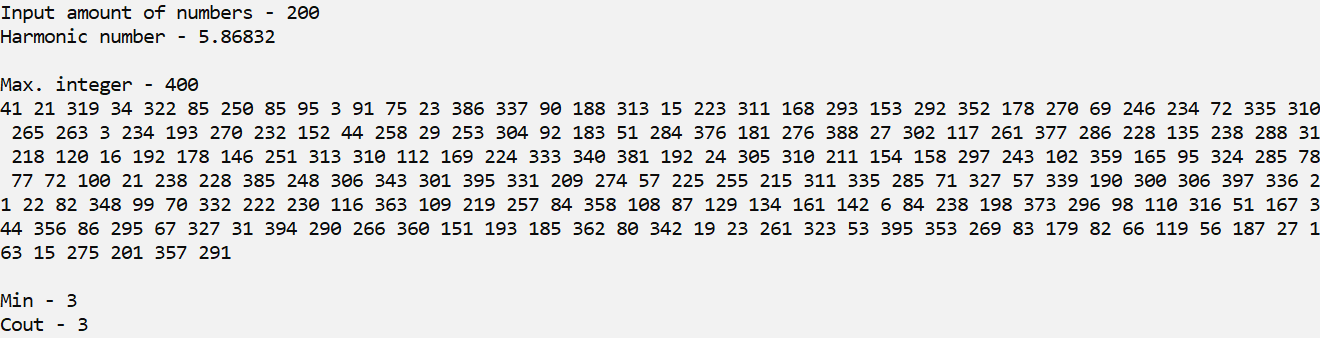


Рисунок 2 – Вычисления для N=200

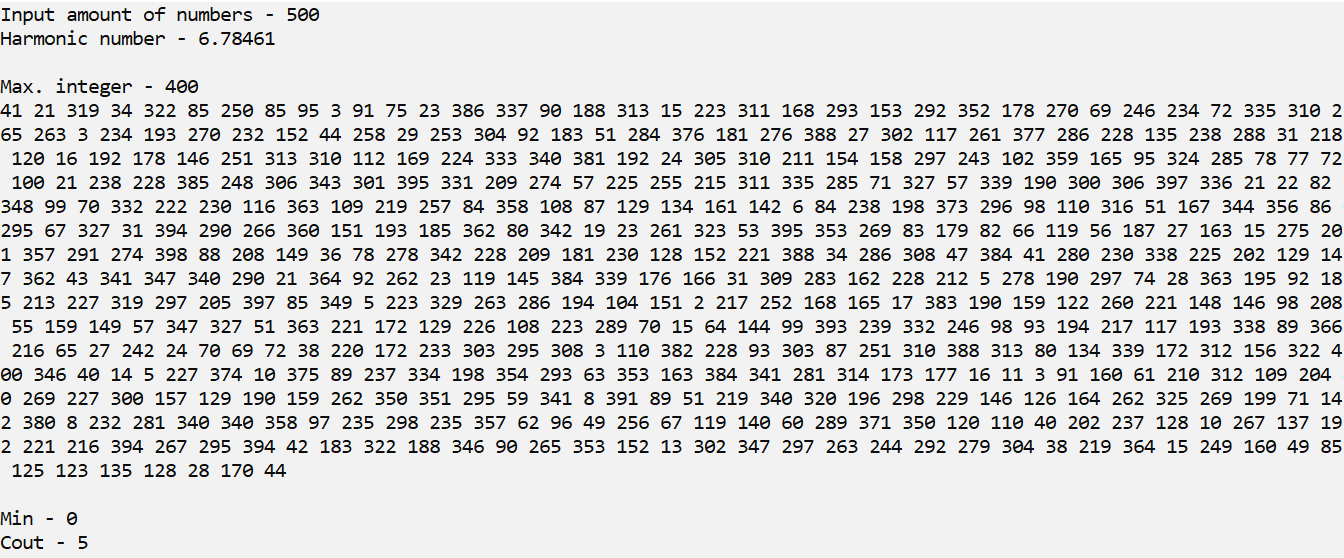


Рисунок 3 – Вычисления для N=500

По единичным результатам вычислений о чём-либо судить трудно, однако при многократном запуске программы для оного и того же значения N можно заметить, что количество операций переприсваивания (count) колеблется около значения n-го гармонического числа.

Вывод

В ходе работы был исследован метод теоретической оценки трудоёмкости алгоритма на основе вычисления гармонического среднего. Был проведён эксперимент на алгоритме поиска минимального числа в массиве, в ходе которого число затраченных операций сравнивалось с их математическим ожиданием. Результаты показали, что для указанного примера n-ое гармоническое число показывает примерное усреднённое значение наиболее вероятного числа затраченных операций.