МІНІСТЕРСТВО ОСВІТИ І НАУКИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ «ХПІ»

Кафедра «Обчислювальна техніка та програмування»

Розрахункове завдання з дисципліни

«Алгоритми та структури даних»

Пояснювальна записка

Розробники

###### Виконав:

студент групи КІТ 119А

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Момот Р.Є./

Перевірив:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Бречко В.А./

Харків 2020

ЗАТВЕРДЖЕНО

КІТ.119А.20205-14 80 01-1 -АЗ

Розрахункове завдання з дисципліни

«Алгоритми та структури даних»

Пояснювальна записка

КІТ.119А.20205-14 80 01-1 -АЗ

Листів \_7\_

Харків 2020

1. Завдання.

**Довгі числа.** Реалізувати предсталення в пам’яті довгих беззнакових цілих чисел. Забезпечити виконання арифметичних операцій множення та ділення таких чисел. Порівняти час виконання зазначених операцій при роботі з даними типу unsigned long та запропонованою реалізацією довгих чисел.

1. Теоретичні відомості по темі завдання.

**Unsigned long int -** це тип даних розширеного розміру для зберігання додатних чисел розміром до 4 байт (32 біт). Змінні типу unsigned long int можуть містити числа від 0 до 4 294 967 295 (232 - 1).

1. Текст програми

#ifndef \_LONGNUMBER\_H\_

#define \_LONGNUMBER\_H\_

#define MAX\_BITS 512

#define \_\_CRTDBG\_MAP\_ALLOC

#include <crtdbg.h>

#define DEBUG\_NEW new(\_NORMAL\_BLOCK, \_\_FILE\_\_, \_\_LINE\_\_)

#define new DEBUG\_NEW

#include <iostream>

#include <string>

#include <chrono>

#include <locale>

using namespace std;

class LongNumber {

char m\_bits[MAX\_BITS];

int m\_nbits;

public:

void clear();

LongNumber operator\*(LongNumber);

LongNumber operator/(LongNumber);

bool operator==(LongNumber);

bool operator<(LongNumber);

bool operator>(LongNumber);

bool operator<=(LongNumber);

LongNumber(unsigned long int k = 0);

LongNumber(const LongNumber&);

~LongNumber();

};

#endif

int main()

{

setlocale(LC\_ALL, "Rus");

cout << "Оператор \*\n";

auto beginClock = chrono::steady\_clock::now();

for (LongNumber i = 2; i < 1000000; i = i \* 2);

auto endClock = chrono::steady\_clock::now();

auto resultClock = chrono::duration\_cast<chrono::nanoseconds>(endClock - beginClock);

cout << "LongNumber: " << resultClock.count() << " нс\n";

beginClock = chrono::steady\_clock::now();

for (unsigned long i = 2; i < 1000000; i = i \* 2);

endClock = chrono::steady\_clock::now();

resultClock = chrono::duration\_cast<chrono::nanoseconds>(endClock - beginClock);

cout << "unsigned long: " << resultClock.count() << " нс\n";

cout << "\n";

cout << "Оператор /\n";

beginClock = chrono::steady\_clock::now();

for (LongNumber i = 1000000; i > 0; i = i / 2);

endClock = chrono::steady\_clock::now();

resultClock = chrono::duration\_cast<chrono::nanoseconds>(endClock - beginClock);

cout << "LongNumber: " << resultClock.count() << " нс\n";

beginClock = chrono::steady\_clock::now();

for (unsigned long i = 1000000; i > 0; i = i / 2);

endClock = chrono::steady\_clock::now();

resultClock = chrono::duration\_cast<chrono::nanoseconds>(endClock - beginClock);

cout << "unsigned long: " << resultClock.count() << " нс\n";

if (\_CrtDumpMemoryLeaks())

cout << "\nУтечка памяти обнаружена." << endl;

else

cout << "\nУтечка памяти отсутствует." << endl;

return 0;

}

LongNumber::LongNumber(unsigned long int value)

{

clear();

if (value == 0)

m\_nbits = 1;

else

while (value > 0)

{

m\_bits[m\_nbits++] = value % 2;

value /= 2;

}

}

LongNumber::LongNumber(const LongNumber& other)

{

memcpy(m\_bits, other.m\_bits, MAX\_BITS);

m\_nbits = other.m\_nbits;

}

LongNumber::~LongNumber()

{

clear();

}

void LongNumber::clear()

{

memset(m\_bits, 0, MAX\_BITS);

m\_nbits = 0;

}

LongNumber LongNumber::operator\*(LongNumber element)

{

LongNumber result;

result.m\_nbits = m\_nbits + element.m\_nbits + 1;

for (int i = 0; i < m\_nbits; i++)

{

char carry = 0;

for (int j = 0; j < element.m\_nbits || carry; j++)

{

char cur = result.m\_bits[i + j] + m\_bits[i] \* (j < element.m\_nbits ? element.m\_bits[j] : 0) + carry;

result.m\_bits[i + j] = cur % 2;

carry = cur / 2;

}

}

while (result.m\_nbits - 1 > 0 && result.m\_bits[result.m\_nbits - 1] == 0)

result.m\_nbits--;

return result;

}

LongNumber LongNumber::operator/(LongNumber element)

{

if (\*this < element || \*this == 0 || element == 0)

return 0;

if (\*this == element)

return 1;

if (element == 1)

return \*this;

int i = 0;

while (element \* i <= \*this)

i++;

return LongNumber(--i);

}

bool LongNumber::operator==(LongNumber element)

{

if (m\_nbits != element.m\_nbits)

return false;

for (int i = 0; i < m\_nbits; i++)

if (m\_bits[i] != element.m\_bits[i])

return false;

return true;

}

bool LongNumber::operator<(LongNumber element)

{

if (\*this == element)

return false;

if (m\_nbits != element.m\_nbits)

return m\_nbits < element.m\_nbits;

for (int i = m\_nbits - 1; i >= 0; i--)

if (m\_bits[i] != element.m\_bits[i])

return m\_bits[i] < element.m\_bits[i];

return false;

}

bool LongNumber::operator>(LongNumber element)

{

return (\*this == element || \*this < element) ? false : true;

}

bool LongNumber::operator<=(LongNumber element)

{

return (\*this < element || \*this == element) ? true : false;

}

1. Результати роботи програми (рис.1).

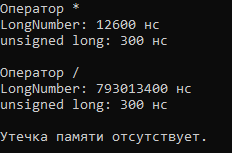


Рис.1 Результат роботи програми

1. Пояснення алгоритму програми

Програма порівнює час роботи розробленого представлення довгих без знакових чисел та типу unsigned long на прикладі операцій множення та ділення. Спочатку виконується замір часу множення на 2 чисел від 2 до 1000000 у циклі для розробленого представлення чисел, а потім виконується такий же самий цикл для типу unsigned long. Після цього починається замір часу для операції ділення на 2 у циклі розробленої реалізації чисел та unsigned long.

1. Висновок

У ході виконання розрахунково-графічного завдання було набуто теоретичні знання та практичні навички роботи з представленням в пам’яті довгих без знакових цілих чисел та типу unsigned long int. Результати роботи програми показують, що операції над типом unsigned long int виконуються швидше, ніж над запропованою реалізацією, бо операції над unsigned long int створені та оптимізовані на найнижчих рівнях машин, на відміну від розробленого представлення.