|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

|  |  |
| --- | --- |
| **ФАКУЛЬТЕТ** | ***ИУК «Информатика и управление»*** |
| **КАФЕДРА** | ***ИУК4 «Программное обеспечение ЭВМ,*** |
|  | ***информационные технологии»*** |

**ЛАБОРАТОРНАЯ РАБОТА №6**

**«Обработка структур»**

**ДИСЦИПЛИНА: «Высокоуровневое программирование»**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил: студент гр. ИУК4-11Б | |  |  | ( | Суриков Н.С | ) |
|  |  |  | (подпись) |  | (Ф.И.О.) |  |
| Проверил: | |  |  | ( | Пчелинцева Н.И. | ) |
|  |  |  | (подпись) |  | (Ф.И.О.) |  |

|  |  |
| --- | --- |
| Дата сдачи (защиты):  Результаты сдачи (защиты): | |
|  | - Балльная оценка:  - Оценка: |

**Цель работы:** приобретение практических навыков процедурного программирования средствами языка С++.

**Задачи:**

1. Изучить тип данных структура.
2. Познакомиться с операциями, предназначенными для работы со структурами.
3. Научиться создавать пользовательское меню для демонстрации заданий лабораторной работы.

**Вариант 20**

**Формулировка общего задания:**

*Задача 1*

Ввести структуру (с полями числитель и знаменатель) для описания понятия рациональное число. Составить и протестировать функцию:

**МАКС (Х, N),**

которая возвращает наибольшее из массива X[N] рациональных чисел.

*Задача 2*

Определить структуру – важнейшие исторические даты.

Ее поля – год, событие.

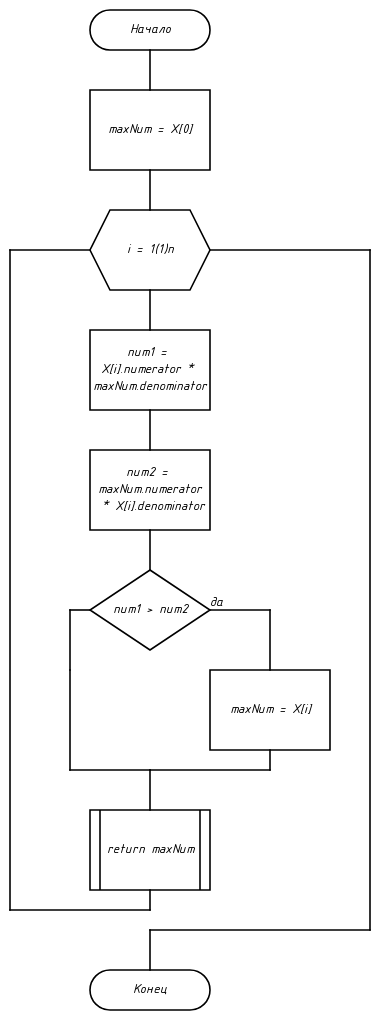
Написать и протестировать функции:

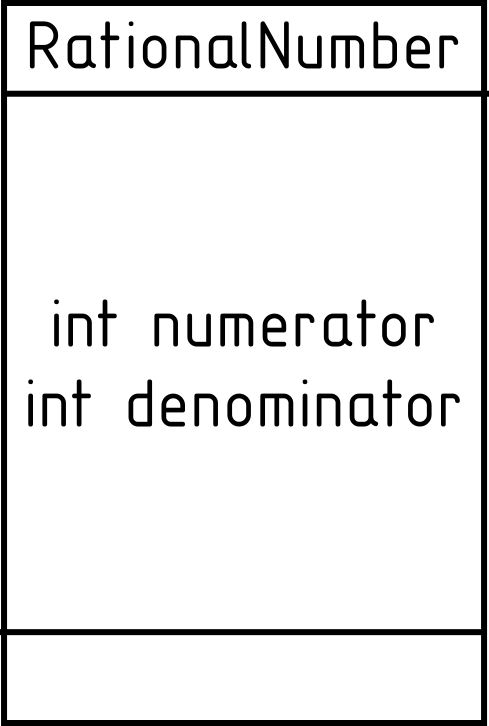
* сортирующие структуры по любому из полей;
* подсчитывающую средний интервал между датами;
* определяющую наиболее часто встречающуюся первую букву в названии события.

*Задача 3*

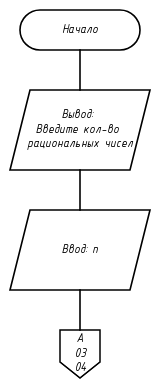
Ввести структуру для описания комплексного числа. Составить и протестировать функцию для умножения комплексных чисел в алгебраической форме.

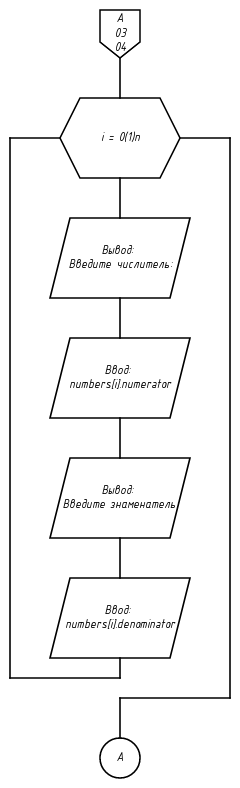
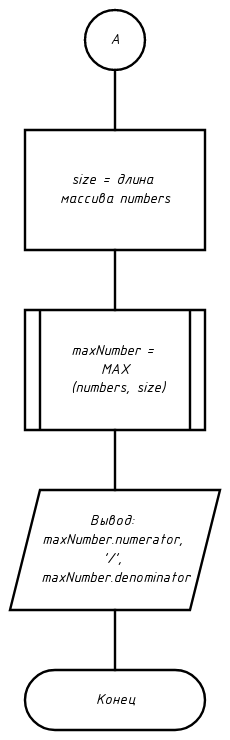
*Блок схема:*

struct RationalNumber MAX(struct RationalNumber X[], int n)

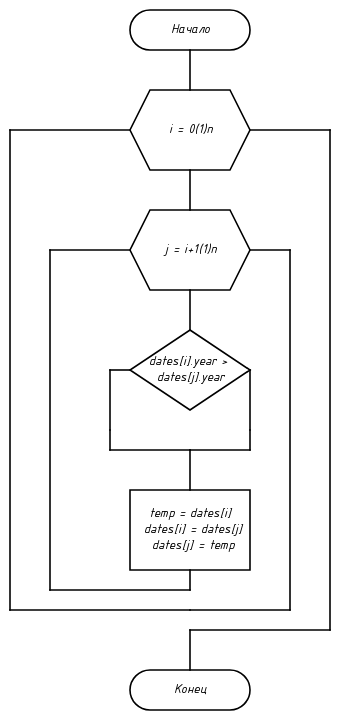
**

void task1()

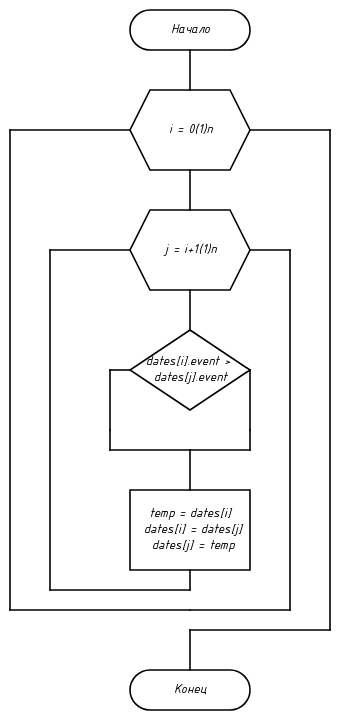
**

* *

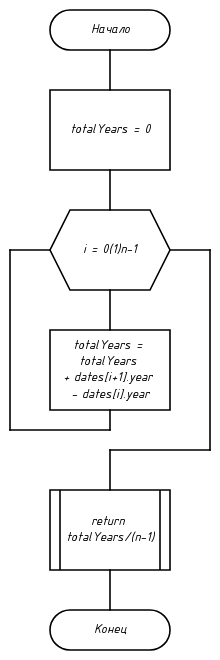
void sortByYear(HistoricalDate dates[], int n)

* *

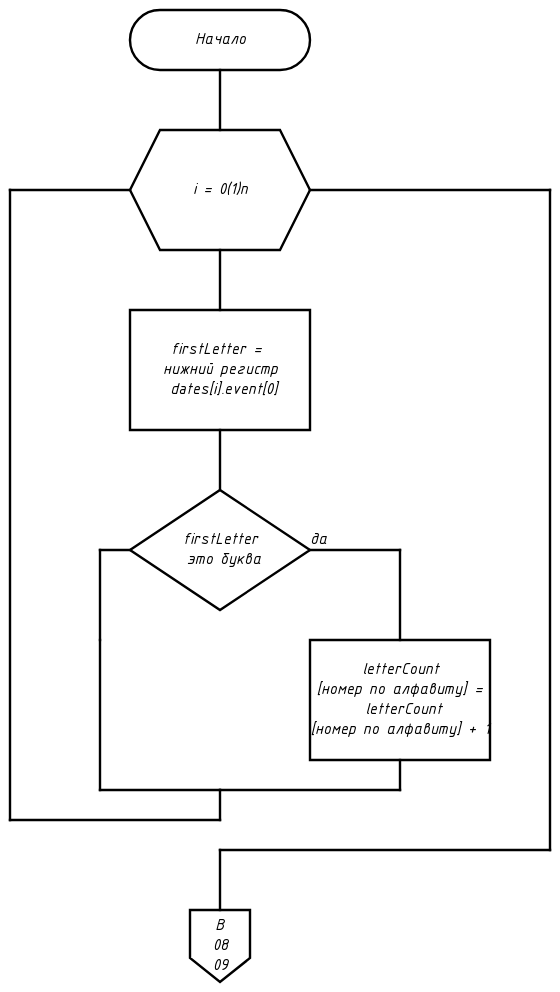
void sortByEvent(HistoricalDate dates[], int n)

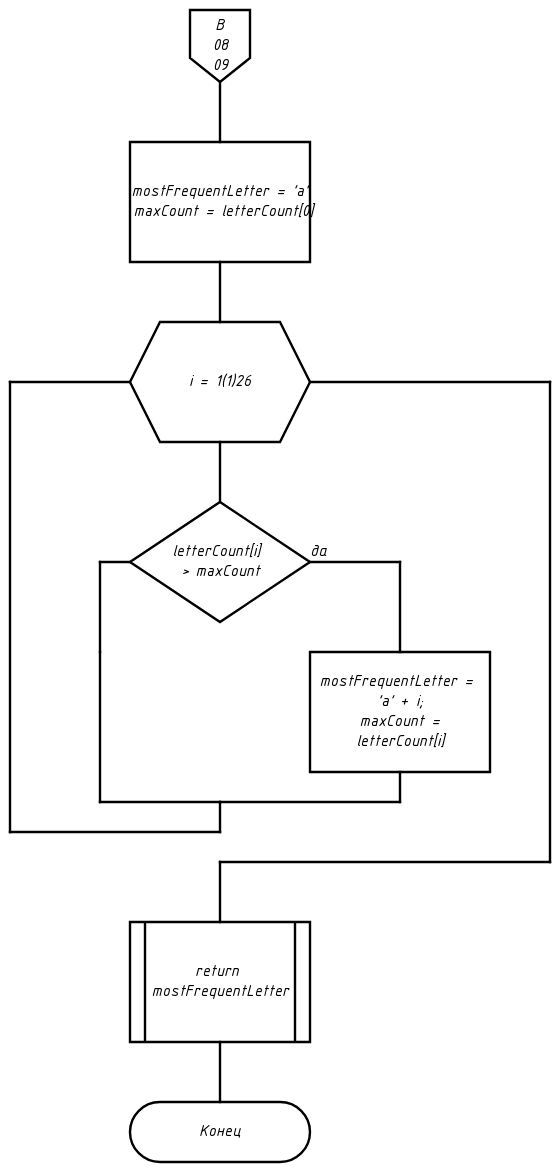
**

float averageInterval(HistoricalDate dates[], int n)

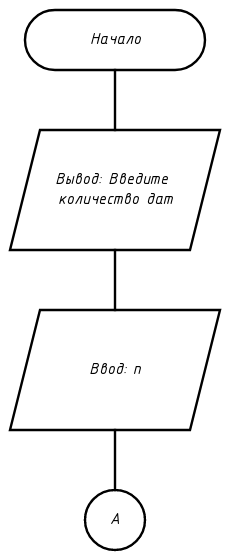
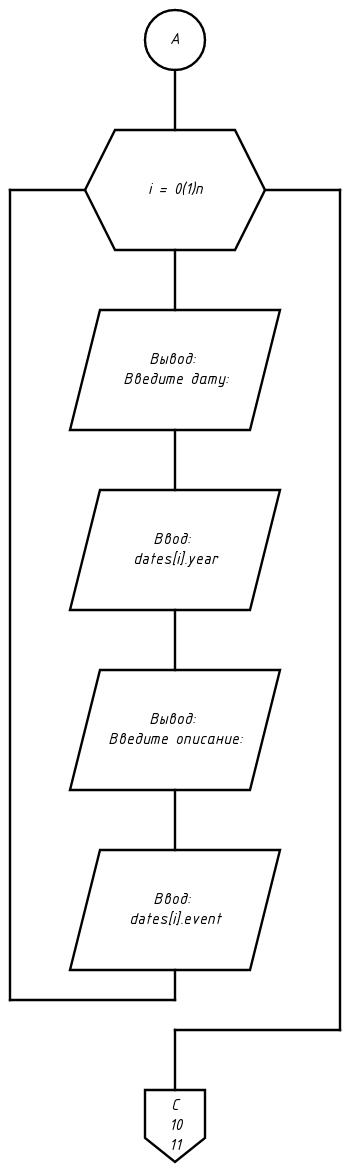
**

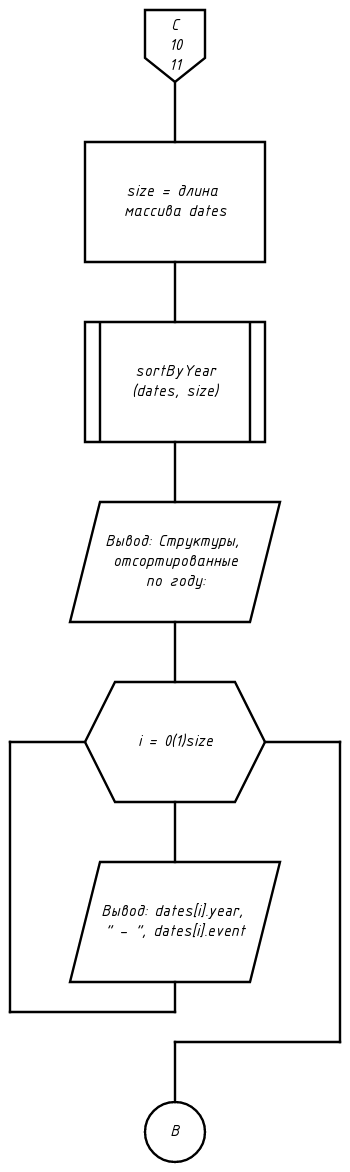
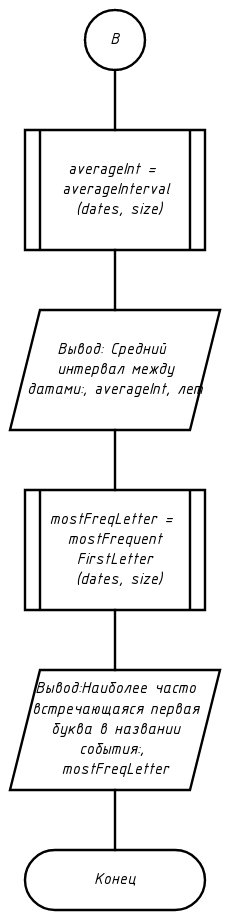
char mostFrequentFirstLetter(HistoricalDate dates[], int n)

**



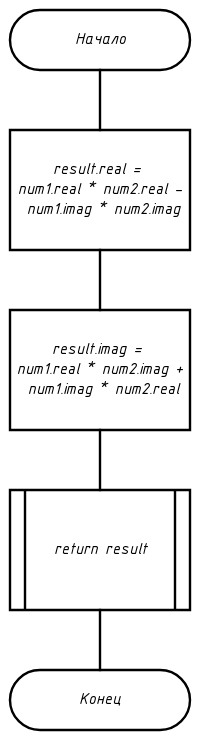
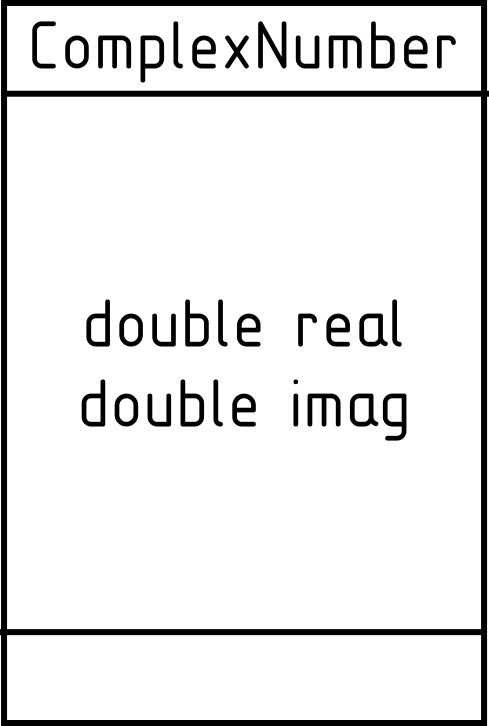
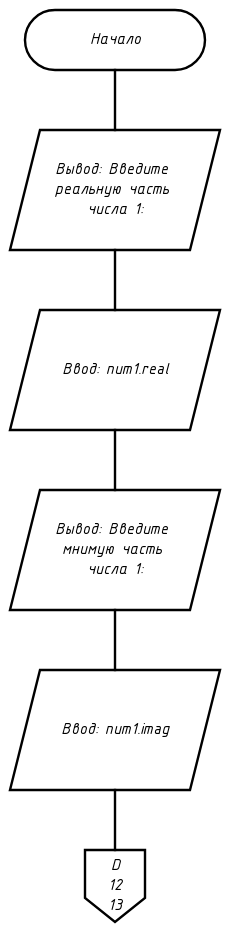
void task2()

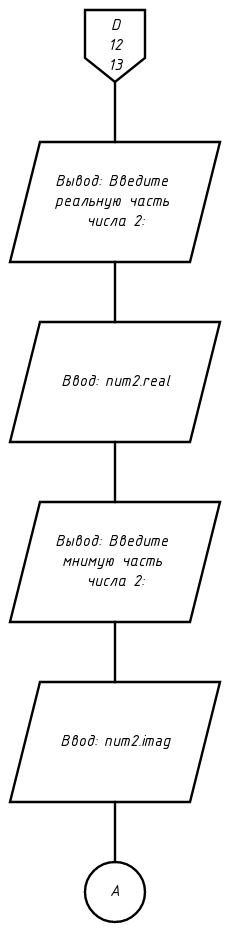
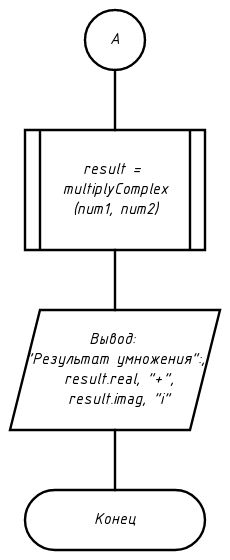
 

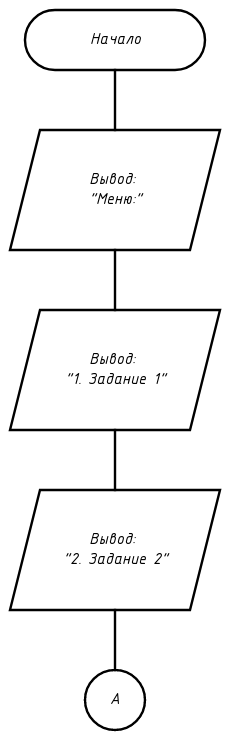
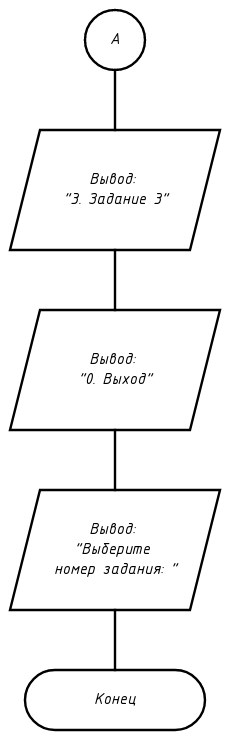
ComplexNumber multiplyComplex(ComplexNumber num1, ComplexNumber num2)

void task3()

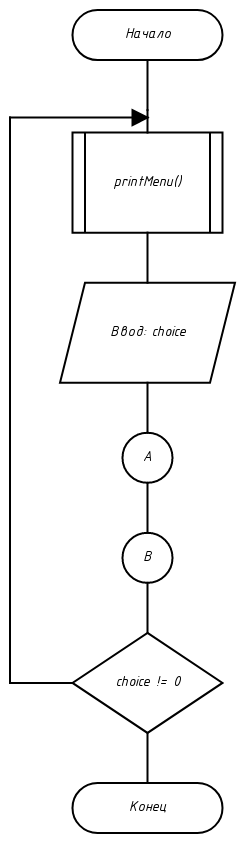
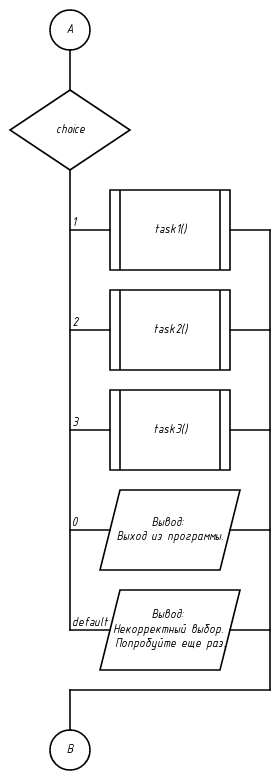
  

* *

void printMenu()

* *

Основная программа

* *

*Листинг программы на C++:*

*#include <iostream>*

#include <cstring>

struct RationalNumber

{

int numerator; // числитель

int denominator; // знаменатель

};

struct HistoricalDate

{

int year; // год

char event[100]; // событие

};

struct ComplexNumber

{

double real; // действительная часть

double imag; // мнимая часть

};

void printMenu();

struct RationalNumber MAX(struct RationalNumber X[], int N);

void sortByYear(HistoricalDate dates[], int N);

void sortByEvent(HistoricalDate dates[], int N);

float averageInterval(HistoricalDate dates[], int N);

char mostFrequentFirstLetter(HistoricalDate dates[], int N);

ComplexNumber multiplyComplex(ComplexNumber num1, ComplexNumber num2);

void task1()

{

std::cout << "Введите количество рациональных чисел" << std::endl;

int n{};

std::cin >> n;

RationalNumber numbers[n];

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

{

std::cout << "Введите числитель: ";

std::cin >> numbers[i].numerator;

std::cout << "Введите знаменатель: ";

std::cin >> numbers[i].denominator;

}

int size = sizeof(numbers) / sizeof(numbers[0]);

struct RationalNumber maxNumber = MAX(numbers, size);

std::cout << maxNumber.numerator << '/' << maxNumber.denominator << std::endl;

}

void task2()

{

std::cout << "Введите количество дат" << std::endl;

int n{};

std::cin >> n;

HistoricalDate dates[n];

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

{

std::cout << "Введите дату: ";

std::cin >> dates[i].year;

std::cout << "Введите описание: ";

std::cin >> dates[i].event;

}

// HistoricalDate dates[] = {

// {1776, "Declaration of Independence of the United States"},

// {1865, "Abolition of slavery in the United States"},

// {1945, "Victory over fascism in World War II"},

// {1917, "October Revolution"},

// {1969, "Moon landing"}};

int size = sizeof(dates) / sizeof(dates[0]);

// Тестирование функции sortByYear

sortByYear(dates, size);

std::cout << "Структуры, отсортированные по году:" << std::endl;

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

{

std::cout << dates[i].year << " - " << dates[i].event << std::endl;

}

// Тестирование функции sortByEvent

sortByEvent(dates, size);

std::cout << "\nСтруктуры, отсортированные по событию:" << std::endl;

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

{

std::cout << dates[i].year << " - " << dates[i].event << std::endl;

}

// Тестирование функции averageInterval

float averageInt = averageInterval(dates, size);

std::cout << "Средний интервал между датами: " << averageInt << " лет" << std::endl;

// Тестирование функции mostFrequentFirstLetter

char mostFreqLetter = mostFrequentFirstLetter(dates, size);

std::cout << "Наиболее часто встречающаяся первая буква в названии события: " << mostFreqLetter << std::endl;

}

void task3()

{

// Создание двух комплексных чисел

ComplexNumber num1, num2;

std::cout << "Введите реальную часть числа 1:";

std::cin >> num1.real;

std::cout << "Введите мнимую часть числа 1:";

std::cin >> num1.imag;

std::cout << "Введите реальную часть числа 2:";

std::cin >> num2.real;

std::cout << "Введите мнимую часть числа 2:";

std::cin >> num2.imag;

// num1.real = 2.5;

// num1.imag = 3.7;

// num2.real = -1.8;

// num2.imag = 4.2;

// Умножение комплексных чисел

ComplexNumber result = multiplyComplex(num1, num2);

// Вывод результата

std::cout << "Результат умножения: " << result.real << " + " << result.imag << "i" << std::endl;

}

int main()

{

int choice;

do

{

// Вывод меню

printMenu();

std::cin >> choice;

// Выполнение выбранного задания

switch (choice)

{

case 1:

task1();

break;

case 2:

task2();

break;

case 3:

task3();

break;

case 0:

std::cout << "Выход из программы." << std::endl;

break;

default:

std::cout << "Некорректный выбор. Попробуйте еще раз." << std::endl;

break;

}

std::cout << std::endl;

} while (choice != 0);

}

void printMenu()

{

std::cout << "Меню:" << std::endl;

std::cout << "1. Задание 1" << std::endl;

std::cout << "2. Задание 2" << std::endl;

std::cout << "3. Задание 3" << std::endl;

std::cout << "0. Выход" << std::endl;

std::cout << "Выберите номер задания: ";

}

struct RationalNumber MAX(struct RationalNumber X[], int n)

{

struct RationalNumber maxNum = X[0];

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

{

int num1 = X[i].numerator \* maxNum.denominator;

int num2 = maxNum.numerator \* X[i].denominator;

if (num1 > num2)

{

maxNum = X[i];

}

}

return maxNum;

}

// Функция для сортировки структур по году

void

sortByYear(HistoricalDate dates[], int n)

{

for (int i = 0; i < n - 1; i++)

{

for (int j = i + 1; j < n; j++)

{

if (dates[i].year > dates[j].year)

{

HistoricalDate temp = dates[i];

dates[i] = dates[j];

dates[j] = temp;

}

}

}

}

// Функция для сортировки структур по событию

void sortByEvent(HistoricalDate dates[], int n)

{

for (int i = 0; i < n - 1; i++)

{

for (int j = i + 1; j < n; j++)

{

if (std::strcmp(dates[i].event, dates[j].event) > 0)

{

HistoricalDate temp = dates[i];

dates[i] = dates[j];

dates[j] = temp;

}

}

}

}

// Функция для подсчета среднего интервала между датами

float averageInterval(HistoricalDate dates[], int n)

{

int totalYears{0};

for (int i = 0; i < n - 1; i++)

{

totalYears += dates[i + 1].year - dates[i].year;

}

return (float)totalYears / (n - 1);

}

// Функция для определения наиболее часто встречающейся первой буквы в названии события

char mostFrequentFirstLetter(HistoricalDate dates[], int n)

{

int letterCount[26] = {0};

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

{

char firstLetter = std::tolower(dates[i].event[0]);

if (std::isalpha(firstLetter))

{

letterCount[firstLetter - 'a']++;

}

}

char mostFrequentLetter = 'a';

int maxCount = letterCount[0];

for (int i = 1; i < 26; i++)

{

if (letterCount[i] > maxCount)

{

mostFrequentLetter = 'a' + i;

maxCount = letterCount[i];

}

}

return mostFrequentLetter;

}

// Функция для умножения комплексных чисел

ComplexNumber multiplyComplex(ComplexNumber num1, ComplexNumber num2)

{

ComplexNumber result;

result.real = num1.real \* num2.real - num1.imag \* num2.imag;

result.imag = num1.real \* num2.imag + num1.imag \* num2.real;

return result;

}

**Вывод**

В ходе данной лабораторной работы я приобрёл практические навыки процедурного программирования средствами языка С++, изучил тип данных структура, познакомился с операциями, предназначенными для работы со структурами и научился создавать пользовательское меню для демонстрации заданий лабораторной работы.

**Литература**

1. Курс лекций доцента кафедры ФН1-КФ Пчелинцевой Н.И.
2. Зырянов, К. И. Программирование на C++: учебное пособие / К. И. Зырянов, Н. П. Кисленко. — Новосибирск: Новосибирский государственный архитектурно-строительный университет (Сибстрин), ЭБС АСВ, 2017. — 129 c. — ISBN 978-5-7795-0817- 9. — Текст: электронный // Электронно-библиотечная система IPR BOOKS: [сайт]. — URL: https://www.iprbookshop.ru/85873.html
3. Фридман, А. Л. Язык программирования C++: учебное пособие / А. Л. Фридман. — 3-е изд. — Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2021. — 217 c. — ISBN 978-5-4497-0920-2. — Текст: электронный // Электронно-библиотечная система IPR BOOKS: [сайт]. — URL: https://www.iprbookshop.ru/102076.html