|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | | | |
|  | |
| Институт информационных технологий (ИТ) | |
| Кафедра инструментального и прикладного программного обеспечения (ИППО) | |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1** | |
| **по дисциплине** | |
| **«Объектно-ориентированное программирование»** | |
|  | |
| Выполнил студент группы ИКБО-13-17 | *Хитров Н.О.* |
| Принял ассистент кафедры ИиППО | *Синицын А.В.* |

|  |  |  |
| --- | --- | --- |
| Лабораторные работы выполнены | «\_\_»\_\_\_\_\_\_\_201\_\_ г. |  |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_201\_\_ г. |  |

Москва 2018

**Оглавление**

[Лабораторная работа №1 3](#_Toc510360201)

[Цель лабораторной работы: 3](#_Toc510360202)

[Текст задания: 3](#_Toc510360203)

[UML-диаграмма: 3](#_Toc510360204)

[Код программы: 4](#_Toc510360205)

[Файл main.cpp: 4](#_Toc510360206)

[Файл Complex.h: 5](#_Toc510360207)

[Файл Complex.c: 5](#_Toc510360208)

[Скриншоты выполнения программы: 6](#_Toc510360209)

[Выводы: 7](#_Toc510360210)

# Лабораторная работа №1

Цель лабораторной работы:

Целью данной лабораторной работы освоить на практике создание многофайловых проектов в языке Си/Си++, познакомиться с директивами условной компиляции.

Текст задания:

1. Напишите программу–калькулятор комплексных чисел. Для реализации необходимо разработать абстрактный тип данных – комплексное число. Программа должна реализовывать арифметические операции над комплексными числами. Программа должна быть представлена в виде многофайлового проекта, все прототипы функций, объявления структур должны быть вынесены в заголовочный файл с соответствующим названием. Всего должно быть три файла: файл с объявлениями, файл реализации и файл с функцией main(), демонстрирующий работу с новым типом данных. Файл с объявлениями должен называться Complex.h, файл с реализацией функций должен называться Complex.с, файл с функцией main() может называться main.с Программа должна обеспечивать удобный интерфейс пользователя для работы с ней.

2. На основе разработанного в предыдущем задании типа данных комплексное число написать программу, которая считывает информацию из файла complex.txt — количество комплексных чисел в переменную n, а сами комплексные числа в массив p. Затем происходит поиск комплексного числа с максимальным модулем в массиве p.

UML-диаграмма:

|  |
| --- |
| Complex |
| +re: double  +im: double |
| +double abs\_complex(Complex \*comp);  +Complex \*plus (Complex \*comp1, Complex \*comp2);  +Complex \*minus (Complex \*comp1, Com plex \*comp2);  +Complex \*multiply (Complex \*comp1, Complex \*comp2);  +Complex \*divide (Complex \*comp1, Complex \*comp2);  +void print\_complex(Complex \*complex);  +int read (FILE \*file, Complex \*complex);  +int calculate (FILE \*input\_file); |

Код программы:

### Файл main.cpp:

#include <stdio.h>  
#include <malloc.h>  
#include "Complex.h"  
  
int main(int argc, const char \*argv[]) {  
 FILE \*input\_file = stdin;  
 if (argc == 1) {  
 printf("=>Input your complexes. Example: (2+3i)/(-1-25i) or (2 - 2i) + (2 - 2i)\n");  
 return calculate((FILE \*) input\_file);  
 }  
 if ((input\_file = fopen(argv[1], "r")) == NULL) {  
 printf("=>>[ERROR] Cant open file %s\n", argv[1]);  
 return -1;  
 }  
 printf("=>>[SUCCESS] File %s is opened!\n", argv[1]);  
 printf("=>Count start:\n");  
  
 int size = 10;  
 Complex \*array = (Complex \*) malloc(sizeof(Complex) \* size);  
 int n = 0;  
  
 while (read(input\_file, array + n) != -1) {  
 if (n - 1 >= size)  
 array = realloc(array, sizeof(Complex) \* size \* 2);  
  
 printf("%d: ", n);  
 print\_complex(array + n);  
 n++;  
 fscanf(input\_file, " ");  
 }  
 printf("=>Count finished: %d\n", n);  
  
 double max = abs\_complex(&array[0]);  
 Complex max\_comp = array[0];  
 for (int i = 1; i < n; ++i) {  
 if(max < abs\_complex(&array[i])){  
 max = abs\_complex(&array[i]);  
 max\_comp = array[i];  
 }  
 }  
 printf("Max: |%lf|\n", max);  
 print\_complex(&max\_comp);  
 free(array);  
 fclose(input\_file);

return 0;  
}

### Файл Complex.h:

#ifndef LAB1\_C\_COMPLEX\_H  
#define LAB1\_C\_COMPLEX\_H  
  
typedef struct complex{  
 double re, im;  
} Complex;  
  
double abs\_complex(Complex \*comp);  
Complex \*plus(Complex \*comp1, Complex \*comp2);  
Complex \*minus(Complex \*comp1, Complex \*comp2);  
Complex \*multiply (Complex \*comp1,Complex \*comp2);  
Complex \*divide (Complex \*comp1, Complex \*comp2);  
void print\_complex(Complex \*complex);  
int read(FILE \*file, Complex \*complex);  
int calculate(FILE \*input\_file);  
#endif //LAB1\_C\_COMPLEX\_H

### Файл Complex.c:

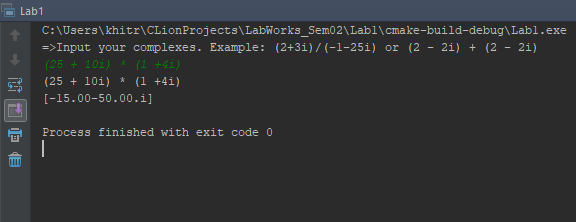
#include <stdio.h>  
#include <math.h>  
#include "Complex.h"  
  
double abs\_complex(Complex \*comp) {  
 return sqrt(comp->re \* comp->re + comp->im \* comp->im);  
}  
  
Complex \*plus(Complex \*comp1, Complex \*comp2) {  
 comp1->re += comp2->re;  
 comp1->im += comp2->im;  
 return comp1;  
}  
  
Complex \*minus(Complex \*comp1, Complex \*comp2) {  
 comp1->re -= comp2->re;  
 comp1->im -= comp2->im;  
 return comp1;  
}  
  
Complex \*multiply (Complex \*comp1,Complex \*comp2) {  
 comp1->re = comp1->re\*comp2->re - comp1->im\*comp2->im;  
 comp1->im = comp1->re\*comp2->im + comp1->im\*comp2->re;  
 return comp1;  
}  
  
Complex \*divide (Complex \*comp1, Complex \*comp2) {  
 double r = comp2->re \* comp2->re + comp2->im \* comp2->im;  
 comp1->re = (comp1->re \* comp2->re + comp1->im \* comp2->im) / r;  
 comp1->im = (comp1->im \* comp2->re - comp1->re \* comp2->im) / r;  
 return comp1;  
}  
  
void print\_complex(Complex \*complex){  
 printf("[%.2lf%+.2lf.i]\n", complex->re, complex->im);  
}  
  
int read(FILE \*file, Complex \*complex) {  
 char op;  
 int res;

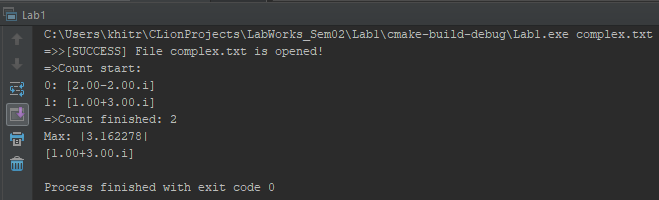
if (

(res =fscanf(file, "(%lf %c %lfi)", &complex->re, &op, &complex->im) )

!= EOF && res == 3) {  
 complex->im \*= op == '-' ? -1 : 1;  
 return 1;  
 }  
 return -1;  
}  
  
int calculate(FILE \*input\_file){  
 Complex complex1;  
 Complex complex2;  
 char operator;  
 char op1, op2;  
 if ( fscanf(input\_file, "(%lf %c %lfi) %c (%lf %c %lfi)", &complex1.re, &op1, &complex1.im, &operator, &complex2.re, &op2, &complex2.im) < 7)  
 operator = 'e';  
 complex1.im \*= op1 == '-' ? -1 : 1;  
 complex2.im \*= op2 == '-' ? -1 : 1;  
 switch (operator) {  
 case '+':  
 plus(&complex1, &complex2);  
 break;  
 case '-':  
 minus(&complex1, &complex2);  
 break;  
 case '\*':  
 multiply(&complex1, &complex2);  
 break;  
 case '/':  
 divide(&complex1, &complex2);  
 break;  
 default:  
 printf("=>>Input error.\n");  
 return -1;  
 }  
 print\_complex(&complex1);  
 fclose(input\_file);  
}

Скриншоты выполнения программы:





Выводы:

Целью данной лабораторной работы освоить на практике создание многофайловых проектов на языке Си/Си++, познакомиться с директивами условной компиляции.