МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования «Балтийский федеральный университет имени Иммануила Канта»

ОНК «Институт высоких технологий»

ОТЧЁТ О ПРОХОЖДЕНИИ

УЧЕБНО-ТЕХНОЛОГИЧЕСКОЙ (ПРОЕКТНО-ТЕХНОЛОГИЧЕСКОЙ) ПРАКТИКИ

на базе Института физико-математических наук и информационных технологий

Выполнил: Ситников Андрей Вадимович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

студент очной формы обучения 1 курса

направления подготовки 01.03.02 Прикладная математика и информатика

профиль обучения «Информатика и программирование»

Руководитель практики

Асситент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Тарачков М.В.

г. Калининград 2022 г.

**Содержание**

Введение 3

Глава 1. Базовые понятия ООП 4

Глава 2.Задания на практику 9

Глава 3.Выполнение заданий 11

Заключение 17

Список литературы 19

**Введение**

Вид практики – Учебная технологическая (проектно-технологическая) практика (далее Учебная практика).

Цель учебной практики: получение первичных профессиональных умений навыков.

Задачи учебной практики:

⦁ Закрепление и углубление теоретических знаний по программированию;

⦁ Изучение основ объектно-ориентированного программирования в C++;

⦁ Понятие принципов проектирования классов, перегрузки операторов, наследования, инкапсуляции данных, деструкторов классов, правила трёх;

⦁ Приобретение и развитие первичных профессиональных навыков и умений по прикладной математике и информатике

**Глава 1. Базовые понятия ООП**

**Инкапсуляция**

Инкапсуляция – одна из ключевых особенностей объектно-ориентированного программирования. Она включает в себя объединение членов данных и функций внутри одного класса. Объединение похожих членов данных и функций внутри класса вместе также помогает в сокрытии данных.

В общем случае инкапсуляция – это процесс упаковки аналогичного кода в одном месте. В C ++ мы можем объединять элементы данных и функции, которые работают вместе внутри одного класса.

В приведенной в приложении 1 программе функция getArea() вычисляет площадь прямоугольника. Чтобы вычислить площадь, ей нужна длина и ширина. Следовательно, члены данных (lengthи breadth) и функция getArea() хранятся вместе в Rectangle классе.

В приведенном в приложении 2 примере мы вычисляем площадь прямоугольника. Чтобы вычислить площадь, нам нужны две переменные: length и breadth и функция: getArea(). Следовательно, мы объединили эти переменные и функции внутри одного класса с именем Rectangle. Здесь переменные и функции могут быть доступны и из других классов. Следовательно, это не скрытие данных, а только инкапсуляция. Мы просто держим похожие коды вместе.

Стоит отметить, что часто рассматривают инкапсуляцию как скрытие данных, но это не совсем так. Инкапсуляция относится к объединению связанных полей и методов вместе. Это можно использовать для скрытия данных. Инкапсуляция сама по себе не скрывает данные.

В C ++ инкапсуляция помогает нам сохранять связанные данные и функции вместе, что делает наш код более чистым и легким для чтения. Это помогает контролировать модификацию наших членов данных.

Рассмотрим ситуацию в приложении 3, когда мы хотим length, чтобы поле в классе было неотрицательным. Здесь мы можем сделать length переменную закрытой и применить логику внутри метода setAge().

Функции getter и setter обеспечивают доступ только для чтения или записи к нашим членам класса. Это помогает развязать компоненты системы. Так мы можем инкапсулировать код в несколько пакетов. Эти несвязанные компоненты (пакеты) могут разрабатываться, тестироваться и отлаживаться независимо и одновременно. И любые изменения в конкретном компоненте не оказывают никакого влияния на другие компоненты.

Мы также можем добиться скрытия данных с помощью инкапсуляции. В приложении 1, если мы изменим length breadth и переменные на privateor или protected, то доступ к этим полям будет ограничен и они будут скрыты от внешних классов. Это называется скрытием данных.

**Наследование**

Наследование – одна из ключевых особенностей объектно-ориентированного программирования на C++. Это позволяет нам создать новый класс (производный класс) из существующего класса (базового класса).

Производный класс наследует функции базового класса и может иметь собственные дополнительные функции. Например, в приложении 4 класс Dog является производным от Animal класса. Поскольку Dog является производным от Animal, члены Animal доступны Dog.

Наследование – это отношение is-a. Мы используем наследование только в том случае, если между двумя классами присутствует отношение is-a.

Вот несколько примеров:

* Автомобиль – это транспортное средство;
* Оранж – это фрукт;
* Хирург – это врач;
* Собака – это животное.

Модификатор доступа protected особенно актуален, когда речь идет о наследовании C++. Как и private члены, protected члены недоступны вне класса. Однако они могут быть доступны производными классами и классами или функциями друзей. Protected члены нужны, если мы хотим скрыть данные класса, но при этом хотим, чтобы эти данные наследовались его производными классами.

В приложении 5 переменная type является protectedи, таким образом, доступна из производного класса Dog. Мы можем видеть это, поскольку мы инициализировали type в Dog классе с помощью функции setType(). С другой стороны, private переменная color не может быть инициализирована Dog. Кроме того, поскольку ключевое слово protected скрывает данные, мы не можем получить доступ к типу непосредственно из объекта DogAnimal или класса.

Различные способы получения классов известны как режимы доступа. Эти режимы доступа имеют следующий эффект:

* public:  
  Если производный класс объявлен в public режиме состояния, то члены базового класса наследуются производным классом так же, как они есть;
* private:  
  В этом случае все члены базового класса становятся privateчленами производного класса;
* protected:  
  public члены базового класса становятся protected членами производного класса.

**Полиморфизм**

Одной из ключевых особенностей наследования классов является то, что указатель на производный класс совместим по типу с указателем на его базовый класс. Полиморфизм – это искусство использования этой простой, но мощной и универсальной функции.

Пример о классах rectangle и triangle, приведённый в приложении 6, написан с помощью указателей с учетом этой особенности. Функция main объявляет два указателя на Polygon (с именем ppoly1и ppoly2). Им присваиваются адреса rect и trgl, соответственно, которые являются объектами типа Rectangleand Triangle. Такие назначения допустимы, так как оба Rectangle и Triangle являются классами, производными от Polygon.

Разыменование ppoly1 и ppoly2(с ppoly1-> and ppoly2->) допустимо и позволяет нам получить доступ к членам их точечных объектов, но поскольку тип обоих ppoly1 и ppoly2 является указателем на Polygon (а не указателем на Rectangle и не указателем на Triangle), можно получить доступ только к членам, унаследованным от Polygon, а не к производным классам Rectangle и Triangle. Вот почему вышеприведенная программа обращается к area членам обоих объектов с помощью rect и trgl напрямую, а не указателей; указатели на базовый класс не могут получить доступ к area членам.

Член area мог бы быть доступен с указателями Polygon, если area бы был членом Polygon вместо члена его производных классов, но проблема в том, что Rectangle и Triangle реализовать разные версии area поэтому нет ни одной общей версии, которая могла бы быть реализована в базовом классе.

**Глава 2. Задания на практику**

**задача №1 «Рекурсия»**

*Вариант 6.*

Дано слово длины N<100. Получить из него слово, в котором буквы записаны наоборот.

*Задачи во всех вариантах следует выполнять с применением рекурсии. Циклы использовать нельзя.*

**задача №2 «Сортировки»**

*Вариант 6*

Сортировка подсчетом.

Реализовать сортировку и оцените ее сложность.

**задача №3 «Двусвязный циклический список»**

Реализуйте двусвязный циклический список, предусмотрите в нем функции ***добавления элемента, вставки, удаления, обхода списка в обоих направлениях***.

При выполнении задачи опирайтесь на лекцию 19 про односвязный список, а также на справочные ресурсы, например:

<https://prog-cpp.ru/data-dcs/>

Будьте готовы объяснить вашу реализацию.

**задача №4 «Оценка сложности алгоритма»**

***Вариант 6.***

Дана строка длины N. Если в ней есть хотя бы одна гласная, отсортируйте буквы по алфавиту.

Оцените сложность алгоритма в лучшем, среднем и худшем случае. Рассчитайте примерное время работы алгоритма. Запрограммируйте его. Определите реальное время работы (воспользуйтесь классом Timer <https://ravesli.com/urok-129-tajming-koda-vremya-vypolneniya-programmy/>) и сравните с расчетным.

Ввод и вывод данных в расчете не учитывать. Считать, что 0 < N <= 10000.

**задача №5 «Бинарный поиск и бисекция»**

Вариант 6

* Реализуйте алгоритм бинарного поиска, о котором шла речь на лекции. Решите задачу выполнения M (0<M<10000) поисковых запросов над массивом из N (0<N<100000) элементов, используя проход по массиву циклом и бинарный поиск. Сравните время работы этих двух подходов.
* Найдите корень уравнения ln x – 1 = 0 с точностью до 10^-4 с помощью бисекции.

**Глава 3. Выполнение ДОПОЛНИТЕЛЬНЫх Заданий на практику**

**Специально для размещения решений мною был создан публичный репозиторий на платформе GitHub (**[**https://github.com/el3m3nTik/Additionals2sem**](https://github.com/el3m3nTik/Additionals2sem)**)**

**Дополнительная задача №1**

#include <iostream>

#include <windows.h>

#define N 100

void reverse(char S[N],int n, char rS[N],int i)

{

n--;

rS[i] = S[n];

i++;

if (n == 0)

return;

reverse(S, n,rS,i);

}

int main()

{

int i = 0;

char S[N];

std::cin >> S;

int n = strlen(S);

char rS[N];

reverse(S, n, rS, i);

std::cout << rS;

}

**Дополнительная задача №2**

#include <iostream>

void Countsort(int n, int mas[])

{

int m = INT\_MAX;

int M = INT\_MIN;

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

{

if (mas[i] > M)

M = mas[i];

if (mas[i] < m)

m = mas[i];

}

int a;

a = M - m + 1;

int\* c = new int[a];

int y[10];

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

c[i] = 0;

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

{

int k = mas[j]-m;

c[k]++;

}

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

{

c[i] += c[i - 1];

}

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

{

int k = mas[j]-m;

y[c[k] - 1] = mas[j];

c[k]--;

}

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

mas[j] = y[j];

delete[] c;

}

Полное решение находится по ссылке(<https://github.com/el3m3nTik/Additionals2sem/blob/main/add2.cpp>)

**Дополнительная задача №3**

#include <iostream>

struct list

{

int field;

struct list\* next;

struct list\* prev;

};

struct list\* init(int a)

{

struct list\* lst;

lst = (struct list\*)malloc(sizeof(struct list));

lst->field = a;

lst->next = lst;

lst->prev = lst;

return(lst);

}

struct list\* addelem(list\* lst, int number)

{

struct list\* temp, \* p;

temp = (struct list\*)malloc(sizeof(list));

p = lst->next;

lst->next = temp;

temp->field = number;

temp->next = p;

temp->prev = lst;

p->prev = temp;

return(temp);

}

Полное решение находится по ссылке (<https://github.com/el3m3nTik/Additionals2sem/blob/main/add3.cpp>)

**Дополнительная задача №4**

#include <iostream>

#include <chrono>

#include <windows.h>

#define N 10000

class Timer

{

private:

using clock\_t = std::chrono::high\_resolution\_clock;

using second\_t = std::chrono::duration<double, std::ratio<1> >;

std::chrono::time\_point<clock\_t> m\_beg;

public:

Timer() : m\_beg(clock\_t::now())

{}

void reset()

{

m\_beg = clock\_t::now();

}

double elapsed() const

{

return std::chrono::duration\_cast<second\_t>(clock\_t::now() - m\_beg).count();

}

};

Полное решение находится по ссылке (<https://github.com/el3m3nTik/Labs_2sem>)

**Дополнительная задача №5**

#include <iostream>

#include <float.h>

double f(double x)

{

return log(x) - 1;

}

double bisection(double a, double b, double eps)

{

double fa = f(a);

while (107)

{

double x = (a + b) / 2;

if ((abs(x - a) < eps) || (abs(x - b) < eps))

return x;

else if (f(x) \* fa > 0)

a = x;

else

b = x;

}

}

int main()

{

double eps = 1e-4;

std::cout << bisection(0, DBL\_MAX, eps);

}

Полное решение находится по ссылке (<https://github.com/el3m3nTik/Labs_2sem>)

**Заключение**

С++ – язык программирования, который поддерживает такие парадигмы программирования как процедурное программирование, объектно-ориентированное программирование, обобщенное программирование, обеспечивает модульность, раздельную компиляцию, обработку исключений, абстракцию данных, объявление типов (классов) объектов, виртуальные функции. Стандартная библиотека включает, в том числе, общеупотребительные контейнеры и алгоритмы. C++ сочетает свойства как высокоуровневых, так и низкоуровневых языков. В сравнении с его предшественником – языком C, наибольшее внимание уделено поддержке объектно-ориентированного и обобщённого программирования.

Смысл философии языка C++ можно определить выражением "доверять программисту". Например, компилятор не будет вам мешать сделать что-то новое, что имеет смысл, но также не будет мешать вам сделать что-то такое, что может привести к сбою. Это одна из главных причин, почему так важно знать то, что мы не должны делать, как и то, что мы должны делать, создавая программы на языке С++.

⦁ Закрепление и углубление теоретических знаний по программированию;

⦁ Приобретение и развитие первичных профессиональных навыков и умений по прикладной математике и информатике.

⦁ Получение основ практического применения знаний в сфере ООП.

⦁ Получение практических навыков работы с локальными и онлайн репозиториями путём применения платформы Github.

⦁ Реализация контроля версий с помощью систем Git.

⦁ Получение практических навыков, связанных с отладкой и устранением ошибок путём использования IDE Visual Studio 2019.

⦁ Получение теоретических знаний и практических навыков, напрямую связанных с оптимизацией и рационализацией алгоритмов работы программы на языке C++ .

⦁ Применение полученных знаний и навыков для решения не типичных задач.

⦁ Написание, сравнение и использование различных видов сортировки больших массивов элементов для выявление оптимальных методов для дальнейшего использования.

⦁ Создание шаблонов для потенциального использования в последующих проектах

⦁ Решение стандартных задач профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности;

⦁ Структурирование уже имеющихся знаний и выполнение с их использованием различных задач

⦁ Освоение на начальном уровне графических редакторов для создания графических объектов требующихся для решения задач проекта

В течение практики все задачи были выполнены, а цели достигнуты.

**Список литературы**

**Перечень учебной литературы ресурсов сети «Интернет», необходимой для проведения практики**

* Варфоломеева, Т. Н. Структуры данных и основные алгоритмы их обработки : учебное пособие / Т. Н. Варфоломеева. - Москва : ФЛИНТА, 2017. - 159 с. - ISBN 978-5-9765-3691-3. - Текст : электронный. - URL: <https://znanium.com/catalog/product/1860018> (дата обращения: 11.01.2022). – Режим доступа: по подписке.
* Гданский, Н. И. Основы теории и алгоритмы на графах : учебное пособие / Н. И. Гданский. — Москва : ИНФРА-М, 2020. — 206 с. — (Высшее образование: Бакалавриат). - ISBN 978-5-16-014386-6. - Текст : электронный. - URL: <https://znanium.com/catalog/product/978686> (дата обращения: 11.01.2022). – Режим доступа: по подписке.
* Затонский, А. В. Программирование и основы алгоритмизации. Теоретические основы и примеры реализации численных методов: учебное пособие / А.В. Затонский, Н.В. Бильфельд. — 2-е изд. — Москва: РИОР : ИНФРА-М, 2022. — 167 с. — (Высшее образование). — DOI: https: //www.dx.doi.org/10.12737/20468. - ISBN 978-5-369-01195-9. - Текст: электронный. - URL: <https://znanium.com/catalog/product/1860435> (дата обращения: 16.02.2022). – Режим доступа: по подписке.
* Воронцова, Е. А. Программирование на С++ с погружением: практические задания и примеры кода - Москва: НИЦ ИНФРА-М, 2016. - 80 с. ISBN 978-5-16-105159-7. - Текст: электронный. - URL: <https://znanium.com/catalog/product/563294> (дата обращения: 16.02.2022). – Режим доступа: по подписке.

**Перечень ресурсов информационно-телекоммуникационной сети «Интернет»**

* Колдаев, В. Д. Структуры и алгоритмы обработки данных [Электронный ресурс]: учеб. пособие для вузов/ В. Д. Колдаев. - Москва: РИОР; Москва: ИНФРА-М, 2014. - 1 эл. опт. диск (CD-ROM), 294 с.: ил., табл. - (Высшее образование - бакалавриат). - Библиогр.: с. 285. - Лицензия до 23.06.2020 г. - ISBN 978-5-369-01264-2. - ISBN 978-5-16-009012-2: 15100.00 р. Имеются экземпляры в отделах /There are copies in departments: всего /all 2: ЭБС Кантиана(1), ч.з.N1(1) Свободны / free: ЭБС Кантиана(1), ч.з.N1(1)
* Основы программирования. – режим доступа: <http://www.intuit.ru/studies/courses/648/504/info>
* Видеолекции по курсу «Основы программирования». – режим доступа: <http://www.youtube.com/watch?v=pxR3UoO9c9w>
* Сайт «Структуры и алгоритмы». – режим доступа: <http://www.structur.h1.ru/>
* Структуры и алгоритмы обработки данных: Учебное пособие. – режим доступа: <http://window.edu.ru/resource/820/44820>

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования «Балтийский федеральный университет имени Иммануила Канта»

ОНК «Институт высоких технологий»

ДНЕВНИК

УЧЕБНО-ТЕХНОЛОГИЧЕСКОЙ (ПРОЕКТНО-ТЕХНОЛОГИЧЕСКОЙ) ПРАКТИКИ

г. Калининград 2022 г.

**1.Информационная часть**

Королев Борис Сергеевич, студент очной формы обучения 1 курса группы 03\_ПМ\_21\_О\_ИвП направления подготовки 01.03.02 Прикладная математика и информатика профиль обучения «Информатика и программирование» в соответствии с приказом от 22 декабря 2021 г. № 4888ст направляется на учебно-технологическую (проектно-технологическую) практику в Институт физико-математических наук и информационных технологий.

Период практики – с 17.01.2022 г. по 02.07.2022 г.

Руководитель практики от университета – старший преподаватель Тарачков М.В.

Институт физико-математических наук и информационных технологий

Контактный номер телефона +7 (4012) 338 217

Первый заместитель

директора ОНК «ИВТ» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Шпилевой А.А.

**2. Программа практики**

**2.1. План работы**

|  |  |  |
| --- | --- | --- |
| **№**  **п.п.** | **Рабочее место практиканта** | **Продолжительность (в днях)** |
| 1. | Компьютерный класс (аудитория 230а) | 18 |

**2.2. Индивидуальное задание по профилю подготовки/специальности**

* Пройти инструктаж по технике безопасности.
* Ознакомиться и выполнить задачи на практику.
* Написать отчет по практике.

Руководитель практики

Старший преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Тарачков М.В.

**3. Ход выполнения практики**

|  |  |  |  |
| --- | --- | --- | --- |
| **№**  **п.п.** | **Дата** | **Описание выполненной работы** | **Отметки руководителя практики** |
| 0 | 22.01.2022 | Пройден инструктаж по технике безопасности |  |
| 1 | 22.01.2022 | Задача №1 | 36  36 |
| 2 | 29.01.2022 |
| 3 | 05.02.2022 |
| 4 | 12.02.2022 | Задача №2 |
| 5 | 19.02.2022 |
| 6 | 26.02.2022 |
| 7 | 05.03.2022 | Задача №3 | 36 |
| 8 | 12.03.2022 |
| 9 | 19.03.2022 |
| 10 | 26.03.2022 |
| 11 | 02.04.2022 | Задача №4 |
| 12 | 09.04.2022 | 0 |
| 13 | 16.04.2022 |
| 14 | 23.04.2022 |
| 15 | 30.04.2022 | Задача №5 |
| 16 | 07.05.2022 | 0 |
| 17 | 14.05.2022 |
| 18 | 21.05.2022 |

**3.1 Шкала баллов за выполненные лабораторные работы**

|  |  |  |
| --- | --- | --- |
| **№**  **п.п.** | **Наименование работы** | **Максимально возможное количество баллов** |
| 1 | Задача №1 | 36 |
| 2 | Задача №2 | 36 |
| 3 | Задача №3 | 36 |
| 4 | Задача №4 | 36 |
| 5 | Задача №5 | 36 |

**3.2 Шкала перевода баллов в оценки**

|  |  |  |
| --- | --- | --- |
| **№**  **п.п.** | **Диапазон** | **Оценка** |
| 1 | От 145 до 180 баллов | 5 |
| 2 | От 73 до 144 баллов | 4 |
| 3 | От 1 до 72 баллов | 3 |
| 4 | 0 баллов | 2 |

**4. Отзыв руководителя практики**

Королев Борис Сергеевич, студент очной формы обучения 1 курса группы 03\_ПМ\_21\_О\_ИвП направления подготовки 01.03.02 Прикладная математика и информатика профиль обучения «Информатика и программирование» в соответствии с приказом от 22 декабря 2021 г. № 4888ст направлен на учебно-технологическую (проектно-технологическую) практику в Институт физико-математических наук и информационных технологий.

Период практики – с 17.01.2022 г. по 02.07.2022 г.

Программа практики и индивидуальное задание на практику выполнены. Отчёт по практике сдан и защищён на отчётной конференции.

Студент Королев Борис Сергеевич в процессе прохождения практики справился с поставленными задачами, приобрёл первичные профессиональные навыки и компетенции, в том числе:

* способность решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности;
* способность к разработке и применению алгоритмических и программных решений в области системного и прикладного программного обеспечения.

Учебная практика оценена на оценку 3

Руководитель практики

Ассистент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Тарачков М.В.

« » \_\_\_\_ 2022 г.