МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВВГУ»)

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И АНАЛИЗА ДАННЫХ
КАФЕДРА ИНОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №5

по дисциплине
«Информатика и программирование»

Студент
гр. БИН-25-2 _____ Шумейко И.А

Ассистент
Преподавателя _____ Водяницкий М.В

Владивосток 2025

Задание

Задание 1

Дан список из 10 различных целых чисел. Необходимо найти в нем число 3 и заменить на 30.

Задание 2

Дан список из 5 целых чисел. Необходимо превратить его в список квадратов этих чисел.

Задание 3

Имеется список различных целых чисел. Программа должна найти наибольшее из чисел списка и разделить его на длину списка.

Задание 4

Имеется кортеж из нескольких произвольных элементов. Необходимо этот кортеж отсортировать. Если хотя бы один элемент не является числом, то кортеж остается неизменным.

Задание 5

Имеется словарь товаров в магазине. Необходимо найти товар с минимальной и максимальной ценой.

Задание 6

Имеется список произвольных элементов. Необходимо на основе этого списка создать словарь, где каждый элемент списка будет и ключом, и значением.

Задание 7

Имеется словарь перевода английских слов на русский, где ключ английского слово, значение - русского. Необходимо реализовать программу которая получает на ввод русское слово и результатом выдает перевод на английский.

Задание 8

Реализовать игру Камень-Ножницы-Бумага-Ящерица-Спок. Программа должна запрашивать у пользователя ввод одного из вариантов. Второй вариант случайно генерирует сама программа и возвращает победителя.

Правила игры следующие:

1. Ножницы режут бумагу
2. Бумага покрывает камень
3. Камень давит ящерицу
4. Ящерица отравляет Спока
5. Спок ломает ножницы
6. Ножницы обезглавливают ящерицу

7. Ящерица съедает бумагу
8. Бумага подставляет Спока
9. Спок испаряет камень
10. Камень разбивает ножницы

Задание 9

Дан список слов — например:

```
1 ["яблоко", "груша", "банан", "киви", "апельсин", "ананас"]
```

Необходимо создать новый словарь, где:

- Ключом будет первая буква слова

- Значением - список всех слов, начинающихся с этой буквы

Задание 10

Дан список кортежей, где каждый кортеж содержит имя студента и его оценки, например:

```
1 [("Анна", [5, 4, 5]), ("Иван", [3, 4, 4]), ("Мария", [5, 5, 5])]
```

Необходимо:

1. Создать словарь, где ключ - имя студента, значение - его средняя оценка
2. Найти студента с наибольшей средней оценкой и вывести его имя и средний балл

Содержание

1 Выполнение работы

1.1 Задание 1

Код для данного задания представлен на рисунке 1.

```
std::list<int32_t> i_list {1,2,3,4,5,6,7,8,9,10};
 for(auto & i: i_list){
   if (i == 3){
      i = 30;
   }
 }
```

Рисунок 1 — код для задания 1

Для выполнения задания создадим список чисел i_list, затем пройдемся по списку циклом for-each, используя автоматический вывод типа и при равенстве элемента 3, меняем его на 30.

1.2 Задание 2

Код для данного задания представлен на рисунке 2.

```
std::list<int32_t> i_list {1,2,3,4,5};
for(auto & i: i_list){
   i *= i;
}
```

Рисунок 2 — код для задания 2

Для выполнения задания создадим список i_list из любых чисел и пройдем по нему циклом for-each, при этом умножая каждый элемент на себя.

1.3 Задание 3

Код для выполнения задания представлен на рисунке 3.

```
std::list<int32_t> i_list {1,23,56,83,1,5};
[[maybe_unused]]auto answer = [&i_list](){
    int32_t big_int = 0;
    int32_t count_size = 0;
    for(const auto& i: i_list){
        ++count_size;
        if(i > big_int){
            big_int = i;
        }
    }
    return big_int / count_size;
}();
```

Рисунок 3 — код для задания 3

Для выполения задания создаем список чисел i_list а потом… Мы создаем две переменные: big_int и count_size, затем проходимся циклом по всем элементам в списке при этом конкатенируем count_size и если переменная больше big_int, присваиваем значение этой переменной big_int. После полного прохода по элементам i_list присваиваем перменной answer значение big_int деленное на count_size. Для инициализации answer используем lamda функцию.

1.4 Задание 4

Код для выполнения задания представлен на рисунках 4.1, 4.2, 4.3, 4.4.

```
class Value{
public:
    template<typename T>
    Value(T value):str(std::move(value)){}

    Value(double value):num1(value){}
    Value(int value):num2(value){}
    Value(char value):num3(value){}

    Value(const Value& other){
        str = other.str;
        num1 = other.num1;
        num2 = other.num2;
        num3 = other.num3;
    }
```

Рисунок 4.1 — часть кода для задания 4

```
Value(Value&& other){
    str = std::move(other.str);
    num1 = std::move(other.num1);
    num2 = std::move(other.num2);
    num3 = std::move(other.num3);
}

Value& operator= (const Value& other){
    Value temp(other);
    Swap(temp);
    return *this;
}

Value& operator= (Value&& other){
    Value temp(std::move(other));
    Swap(temp);
    return *this;
}

bool IsStr()const{return str.has_value();}
bool IsDouble()const{return num1.has_value();}
bool IsInt()const{return num2.has_value();}
bool IsChar()const{return num3.has_value();}
bool IsScalar()const{return !IsChar() && !IsStr();}

 const std::string& AsStr()const{return *str;}
 const double& AsDouble()const{return *num1;}
 const int& AsInt()const{return *num2;}
 const char& AsChar()const{return *num3;}
 double AsScalar()const{
     if(IsInt()){
         return *num2;
     }else{
         return *num1;
     }
 }

 bool operator< (const Value& other)const{
     if(other.IsScalar() && other.IsScalar()){
         return AsScalar() < other.AsScalar();
     }
     return false;
 }
```

Рисунок 4.2 — часть кода для задания 4

```
    Value& Swap(Value& other){
        std::swap(str,other.str);
        std::swap(num1,other.num1);
        std::swap(num2,other.num2);
        std::swap(num3,other.num3);
        return *this;
    }

private:
    std::optional<std::string> str;
    std::optional<double> num1;
    std::optional<int> num2;
    std::optional<char> num3;
};

std::ostream& operator<<(std::ostream& os, const Value& val){
    if(val.IsStr()){
        os << val.IsStr();
    }else if(val.IsDouble()){
        os << val.AsDouble();
    }else if(val.IsInt()){
        os << val.AsInt();
    }else{
        os << val.AsChar();
    }
    return os;
}
```

Рисунок 4.3 — код для задания 4

```
    std::vector<Value> my_vector{1,2.3,3.2,5,1,4};

    bool IsOnlyScalar = true;
    for(const auto& val: my_vector){
      if(!val.IsScalar()){
        IsOnlyScalar = false;
        break;
      }
    }

    if(IsOnlyScalar && my_vector.size() != 1){
      bool is_swapped = true;
      while(is_swapped){
        is_swapped = false;
        for(size_t i = 1; i != my_vector.size();++i){
          if(my_vector.at(i) < my_vector.at(i-1)){
            my_vector[i].Swap(my_vector[i-1]);
            is_swapped = true;
          }
        }
      }
    }

    for(const auto& val: my_vector){
      std::cout << val << ' ';
    }
  }
```

Рисунок 4.4 — часть кода для задания 4.

Для выполнения работы создадим класс Value в котором будем хранить 4 переменных через std::optional, определяем функции, которые проверяют содержится ли в этих переменных какие-то числа или строки или символ, и методы, которые будут возвращать ссылки на хранящееся значение, затем создадим вектор с Value и заполним его значениями. После этото проходимся циклом по всем значениям вектора my_vector и, если метод IsScalar возвращает для всех значений true, сортируем векрот пузырьковой сортировкой, затем выводим значения в поток.

1.5 Задание 5

Код для выполнения задания представлен на рисунке 5.

```
struct Stats{
    std::string name;
    int32_t cost;
};

struct Comparator{
    bool operator()(const auto& lhs, const auto& rhs){
        return lhs.cost < rhs.cost;
    }
};

void F5(){
    std::vector<Stats> products{
        {"Булочка",32},{"Пельмень",54},{"Газировочка",15},{"Кулич",1},{"Огурец",6}
    };
    std::ranges::sort(products,Comparator{});
    std::pair<std::string,std::string> min_max = {products.begin()-
>name,products.rbegin()->name};
}
```

Рисунок 5 — код для задания 5

Для выполнения задания создадим структуру Stats с именем и ценой продукта, затем определим условие Comparator при котором объект Stats меньше другого, если меньше значение цены. Наконец создаем вектор продуктов products и сортируем его, используя Comparator. Возьмем значение с начала вектора и с конца, это будут соответственно продукты с меньшей и большей ценой.

1.6 Задание 6

Код для выполнения задания представлен на рисунке 6.

```
std::list<std::string> words;
std::map<std::string,std::string> words_map;

for(const std::string& str: words){
    words_map[str] = str;
}
```

Рисунок 6 – код для задания 6

Для выполнения задания имеем заготовленный список слов words, создадим карту с ключами - словом, значенимем - то же словом и спомощью цикла заполним одинаковыми словами.

1.7 Задание 7

Код для выполнения задания представлен на рисунке 7.

```
std::string elem_name;
std::map<std::string,std::string> translator{};

std::cin >> elem_name;
if(translator.contains(elem_name)){
    std::cout << translator.at(elem_name);
}else{
    std::cout << "Нет такого слова";
}
```

Рисунок 7 – код для задания 7

Для выполнения задания создадим карту с ключами - словами на русском, значениями - их переводом. Затем примем ввод слова с клавиатуры, проверим есть ли такое слово в словаре, и либо выведем слово, либо строчку, которая означает ошибку ввода, то есть — "Нет такого слова".

1.8 Задание 8

Код для выполнения задания представлен на рисунках 8 и 9.

```
std::string first_partner;
std::unordered_map<std::string,std::set<std::string>> game_rules{
    {"Камень",{"Ящерица","Ножницы"}},
    {"Ножницы",{"Бумага","Ящерица"}},
    {"Бумага",{"Камень","Спок"}},
    {"Ящерица",{"Бумага","Спок"}},
    {"Спок",{"Камень","Ножницы"}},
};
std::cin >> first_partner;
if(!game_rules.contains(first_partner)){
    std::cout << "Такого игрока нет" << std::endl;
    return;
}
```

Рисунок 8 — код для задания 8

```
std::string second_partner = [&](){
    uint32_t counter = 0;
        uint32_t random = static_cast<uint32_t>(std::rand()) % static_cast<uint32_t>(game_rules.size());

    std::string partner;

    for(const auto& key: std::views::keys(game_rules)){
        if(counter == random){
            partner = key;
        }else{
            ++counter;
        }
    }
    return partner;
}();

if(game_rules.at(first_partner).contains(second_partner)){
    std::cout << first_partner << " Побеждает " << second_partner;
}else if(game_rules.at(second_partner).contains(first_partner)){
    std::cout << second_partner << " Побеждает " << first_partner;
}
```

Рисунок 9 — код для задания 8

Для выполнения задания создадим карту с правилами игры, где ключом будет имя игрока, который побеждает любого из списка значений. Затем примем ввод имени с клавиатуры в first_partner и проверим, существует ли такой игрок, если нет - выведем ошибку, если существует, то возьмём остаток от деления функции rand на количество игроков, так мы пройдемся циклом по именам и выберем второго случайного игрока и сохраним его в переменной second_partner. Наконец проверим есть ли в списках first_partner second_partner и наоборот, игрок в списке которого есть его партнёр выигрывает. Выводим нужный нам результат.

1.9 Задание 9

Код для выполнения задания представлен на рисунке 10.

```
std::list<std::string> words{"Коктель","Кино","Бандура","Дора"};
std::map<int8_t,std::list<std::string>> words_map;

for(const std::string& str: words){
    words_map[*str.begin()].push_back(str);
}
```

Рисунок 10 — код для задания 9

Для выполнения задания сохраним список слов в words, создадим карту words_map где ключом бубет буква, а значением список слов и при помощи цикла for-each заполним карту, беря первую букву слова и добавляя слово в список.

1.10 Задание 10

Код для выполнения задания представлен на рисунке 11.

```
struct StudentMarks{
    std::string name;
    std::vector<int32_t> marks;
    int32_t AvarageScore(){
        int32_t score;
        for(const auto& m: marks){
            score += m;
        }
        return score / static_cast<int32_t>(marks.size());
    }
};

void F10(){
    std::vector<StudentMarks> Students_{{"Иван",{4,5,4}},{"Саша",{4,5,4}},{"Настя",{4,5,4}}};
    auto good_student = Students_.begin();
    for(auto it = std::next(good_student); it != Students_.end();++it){
        if(good_student->AvarageScore()< it->AvarageScore()){
            good_student = it;
        }
    }
        std::cout << good_student->name << " получила наивысший средний бал: " << good_student->AvarageScore();
}
```

Рисунок 11 — код для задания 10

Для выполнения задания создадим структуру StudentMarks, которая будет содержать имя, оценки и функцию для вывода среднего бала AvarageScore. Затем создаем вектор StudentMarks с именами и оценками студентов, сохраняем итератор на первый элемент вектора и проходимся циклом по следующим и сравниваем их, наибольший по среднему балу оставляем в переменной good_student. В конце выводим имя и средний бал лучшего студента.