Институт информационных технологий

Кафедра: Математическое и программное обеспечение ЭВМ

Дисциплина: Объектно-ориентированное программирование

ЛАБОРАТОРНАЯ РАБОТА № 5

Тема: Взаимодействие классов

Выполнил:

студент гр. 1ПИб-02-2оп-23

Кринкин Олег Алексеевич

Проверил:

ассистент Матевосян Ремик Артурович

ЗАДАНИЕ

1. Ознакомьтесь с УМП по ООП часть 2 раздел VII.
2. Модифицируйте контейнерный класс, реализованный по заданию ЛР2, используя шаблоны определения класса (обязательно) и шаблоны определения функции (по желанию).
3. Проверьте работоспособность АТД на тестовом наборе данных.

Примечания:

* Обязательно сделать шаблонным класс контейнер.
* Шаблоны определения функции на свое усмотрение.

Контрольные вопросы:

1. Для чего используется ключевое слово template?
2. В чем заключаются особенности параметров по умолчанию для шаблонов?
3. Для чего используются шаблоны функций?
4. Назовите разновидности дружественных функций шаблонного класса.
5. Дайте определение специализации шаблона.

ХОД РАБОТЫ

1. Создан новый заголовочный файл stats.h, в котором определён класс Stats с полями: список, количество подключенных клиентов, а также компонентные функции согласно заданию.

|  |
| --- |
| class Stats {  private:  std::map<const Router\*, int> routers;  friend class Tests;  int clients\_total;  public:  Stats();  ~Stats();  void to\_template(ServerRoom<Router>\*) const;  void to\_map(ServerRoom<Router>\*);  void to\_map(Router\*, int);  void increase(Router\*, int);  void decrease(Router\*, int);  void remove(Router\*);  void print() const;  int get\_total() const;  void clear();  }; |

1. Все необходимые компотные функции были реализованы в файле stats.cpp. Их работа основана на вызове различных методов других ранее описанных классов (Router и ServerRoom). Пример одной из компонентных функций:

|  |
| --- |
| void Stats::print() const {  /\*  \* Функция выводящая статистику на экран по списку объектов Router  \*/  for (const auto& pair : routers) {  std::cout << "Router address: " << pair.first->get\_address() << ", count: " << pair.second << std::endl;  }  }; |

1. Дли проверки правильной работы написанных функций был разаботан ряд тестов, которые производят некоторую работу с функциями и проверяют соответствие ожидаемому результату:

|  |
| --- |
| const char\* stats() {  Stats\* stats = new Stats();  ServerRoom<Router>\* room = new ServerRoom<Router>();  for (int i = 0; i < 3; i++) {  room->add(\*r\_units[i]);  }  stats->to\_map(room);  if (stats->get\_total\_devices() == 0) return errors[26];  stats->clear();  if (stats->get\_total\_devices() != 0) return errors[27];  delete stats;  stats = new Stats();  stats->to\_map(r\_units[0], 3);  if (stats->get\_total\_devices() == 0) return errors[26];  stats->increase(r\_units[0], 3);  if (stats->routers[r\_units[0]] != 6) return errors[28];  stats->decrease(r\_units[0], 5);  if (stats->routers[r\_units[0]] != 1) return errors[29];  stats->remove(r\_units[0]);  if (stats->get\_total\_devices() != 0) return errors[30];  if (stats->get\_total() != 0) return errors[31];  return errors[0];  } |

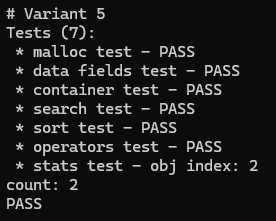
РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ РАБОТЫ

Результат исполнения программы

Используя некоторую систему автоматических «юнит» тестов, которая выполняет несколько тестов:

* выделение и освобождение памяти классами;
* заполнение и хранение данных в полях классов;
* хранение объектов классов в классе-контейнере;
* поиск объекта в контейнере;
* сортировка объектов;
* работа операторов;
* работа статистики.

получен следующий результат:



ЗАКЛЮЧЕНИЕ

В ходе выполнения лабораторной работы были освоены навыки перегрузки операторов как унарных, так и бинарных, как компонентными функциями, так и внешними дружественными. Также объявленные для класса контейнера операторы были применены на практике при алгоритме сортировки и вывода на экран.