ЛАБОРАТОРНАЯ РАБОТА №5

Курс «Объектно-ориентированное программирование»

Тема: Шаблоны. Библиотека STL.

Цель: Получить навыки обобщенного программирования в C++, научиться использовать библиотеку STL для решения различных практических задач.

Темы для предварительной проработки [УСТНО]**:**

- Шаблонные функции и классы.
- Библиотека STL. Контейнеры. Итераторы.
- Библиотека STL. Функторы, алгоритмы.

Индивидуальные задания [КОД] **:**

1. Написать шаблонный класс `DataManager<T>` для специфической работы с набором однотипных данных (максимальная вместимость равна 64 элементам). В набор можно добавлять данные (метод `push(T elem)` для добавления одного элемента и метод `push(T elems[], int n)` для добавления группы из n элементов), считывать без извлечения (метод `T peek()`) и извлекать (метод `T pop()`) по некоторым алгоритмам (в соответствии с вариантом, приложение А). Если набор заполнен на 100% и поступает команда добавления нового элемента(ов), то данные полностью выгружаются (дописываются) в специальный файл dump.dat, а сам массив очищается и новые данные записываются уже в обновленный набор. Если из массива удаляется последний элемент, то он заполняется ранее выгруженными в файл данными (если файл не пуст).

   Необходимо также реализовать явную специализацию шаблонного класса для символьного типа. В ней надо запрограммировать следующее: при добавлении символа в набор все знаки пунктуации должны автоматически заменяться на символ подчеркивания; добавить методы `char popUpper()` и `char popLower()`, которые позволяют при извлечении символа из набора привести его к верхнему или нижнему регистру, соответственно.

   В функции `main()` продемонстрировать применение шаблонного класса `DataManager` для типов `int`, `double` и `char`. Элементы контейнера должны выводиться на консоль с помощью `std::ostream_iterator`.
2. Написать код для чтения произвольного текстового файла и вывода на экран всех слов размером более 3 букв, встречающихся в нем не менее 7 раз, в порядке убывания частоты встречаемости (*приложение А*). В качестве разделителей слов рассматривать пробел, точку, запятую, тире, двоеточие, восклицательный знак, точку с запятой. Использовать контейнер std::map.
3. Создать класс книги Book, в котором хранится название, автор и год издания книги. В главной функции создать коллекцию книг (приложение А). Продемонстрировать сортировку книг по автору (первичный ключ) и названию (вторичный ключ). Продемонстрировать поиск в коллекции: найти все книги, год издания которых находится в указанном диапазоне. Использовать контейнер std::vector и функторы.

4. Подсчитать и вывести на консоль количество всех книг новее 2009 года, используя *только* стандартные алгоритмы и функторы STL (подсказка: `std::count_if`, `std::greater<>`, `std::bind2nd`).
5. Написать шаблонный класс кэша данных `Cache<T>` с методом добавления элемента в кеш `put(T elem)` и его аналогом – оператором +=, а также методом проверки наличия элемента в кеше `bool contains(T elem)`. Написать явную специализацию шаблона для типа `std::string` с такими нюансами: метод `contains()` должен возвращать true по совпадению первого символа строки; метод `add()` должен генерировать исключение, если в кеше уже есть 100 строк. В главной функции инстанцировать `Cache` с типами `int` и `std::string`, добавить в каждый несколько элементов и продемонстрировать для каждого работу метода `contains()` (см. *Приложение А*).
6. Модифицировать код игры «Блек-джек» из лабораторной работы № 4: использовать библиотеку STL для работы с коллекциями объектов.

Контрольные вопросы [ОТЧЕТ]**:**

1. Что такое шаблонная функция? Шаблонный класс?
2. Как компилятор работает с шаблонами? Статические члены шаблонных классов.
3. Что означает явная специализация шаблона? Частичная специализация шаблона?
4. Что означает идиома SFINAE?
5. Кратко опишите типы контейнеров STL.
6. Чем отличаются контейнеры vector, deque и list? Как они реализованы в STL?
7. Кратко опишите типы итераторов STL и методологию работы с ними.
8. Что такое ассоциативный массив? Кратко опишите контейнеры map, multimap, set.
9. Кратко опишите адаптеры контейнеров stack, queue, priority_queue.
10. Что такое функтор? Где и как функторы используются в библиотеке STL?
11. Что означает аббревиатура RAII? Что такое умный указатель?

Рекомендуемые источники:

[1] Страуструп Б. Язык программирования C++ / Б. Страуструп. – СПб.; М.: Невский диалект; Бином, 2015. – 1136 с.

[2] Лафоре Р. Объектно-ориентированное программирование в C++ / Р. Лафоре. – СПб.: Питер, 2013. – 928 с.

[3] Прата С. Язык программирования C++ / С. Прата; пер. с англ. Ю. Н. Артеменко. – Лекции и упражнения, 6-е изд.: Пер. с англ. – М.: Вильямс, 2012. – 1248 с.

[4] Дейтел Х. М. Как программировать на C++ / Х. М. Дейтел, П. Дж. Дейтел. – М.: Бином, 2000. – 1024 с.

[5] Мейерс С. Эффективное использование STL / С. Мейерс. – СПб: Питер, 2002. – 224 с.

[6] Мейерс С. Эффективный и современный C++: 42 рекомендации по использованию C++11 и C++14 / С. Мейерс. – М.: Вильямс, 2016. – 304 с.

[7] Александреску А. Современное проектирование на C++: обобщенное программирование и прикладные шаблоны проектирования / А. Александреску. – М.: Вильямс, 2008. – 336 с.

Приложение А. Некоторые пояснения к заданиям и фрагменты кода.

Задание 1.

Вариант 1.
push(): данные пишутся на первое свободное место в наборе;
peek(): возвращает предпоследний элемент в наборе или 0, если элементов в наборе меньше 2;
pop(): извлекает средний элемент из набора (если элементов четное число, то первый элемент слева от центра раздела набора).

Вариант 2.
push(): данные пишутся в начало набора, остальные смещаются вправо;
peek(): возвращает второй элемент в наборе или 0, если элементов в наборе меньше 2;
pop(): извлекает последний элемент.

Вариант 3.
push(): данные пишутся на первое свободное место в наборе, начиная с конца;
peek(): возвращает центральный элемент в наборе или 0, если число элементов четно;
pop(): извлекает первый элемент.

Вариант 4.
push(): данные пишутся на первое свободное место в наборе;
peek(): возвращает второй элемент в наборе или 0, если элементов в наборе меньше 2;
pop(): извлекает первый элемент.

Вариант 5.
push(): данные пишутся в начало набора, остальные смещаются вправо;
peek(): возвращает центральный элемент в наборе или 0, если число элементов четно;
pop(): извлекает средний элемент из набора (если элементов четное число, то первый элемент слева от центра раздела набора).

Вариант 6.
push(): данные пишутся на свое место в наборе (данные должны упорядочиваться по возрастанию);
peek(): возвращает предпоследний элемент в наборе или 0, если элементов меньше 2;
pop(): извлекает последний элемент.

Вариант 7.
push(): данные пишутся на свое место в наборе (данные должны упорядочиваться по убыванию);
peek(): возвращает второй элемент в наборе или 0, если элементов в наборе меньше 2;
pop(): извлекает первый элемент.

Вариант 8.
push():данные пишутся на свое место в наборе (данные должны упорядочиваться по возрастанию);
peek(): возвращает второй элемент в наборе или 0, если элементов в наборе меньше 2;
pop(): извлекает первый элемент.

Вариант 9.
push():данные пишутся на свое место в наборе (данные должны упорядочиваться по убыванию);
peek(): возвращает последний элемент;
pop(): извлекает предпоследний элемент или последний, если элементов в наборе меньше 2.

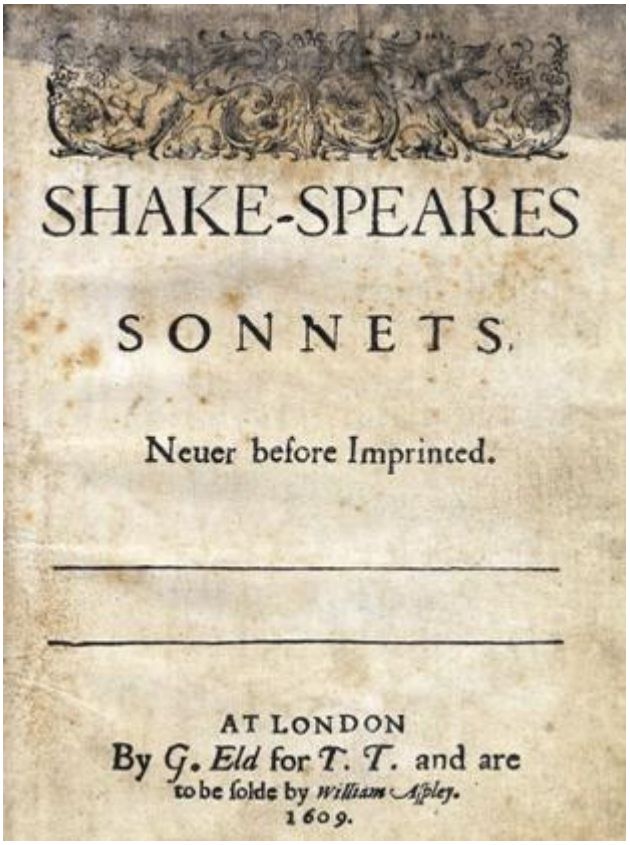
Вариант 10.
push(): данные пишутся на первое свободное место в наборе, начиная с конца;
peek(): возвращает самый большой элемент в наборе;
pop(): извлекает последний элемент.

Пример кода в функции main():

```
DataManager<int> manager;
manager.push(-9);                  // уже в наборе 1 элемент

int a[60] = {0};
manager.push(a, 60);               // уже в наборе 61 элемент

int x = manager.peek();            // узнаем последний элемент (без извлечения)

for (size_t i = 1; i < 15; ++i)
{
       manager.push(i);            // здесь на четвертой итерации должна
}                                  // произойти выгрузка 64 элементов в файл dump.dat
x = manager.pop();                 // сейчас в наборе 11 элементов
                                   // после pop() будет 10

             // кстати, одновременное получение и удаление элемента из контейнера
             // чревато проблемами. Подумайте, какими?

DataManager<char> char_manager;    // явная специализация шаблона для char
char_manager.push('h');
char_manager.push('e');
char_manager.push('l');
char_manager.push('l');
char_manager.push('o');

char ch = char_manager.popUpper();       // этот метод есть только для char
std::cout << ch << std::endl;
```

Задание 2.
На примере файла с 20 сонетами Шекспира Sonnets.txt вывод на консоль должен быть таким:

```
thou 55
that 33
with 27
thee 25
when 16
thine 15
time 15
your 15
this 13
from 13
which 13
beauty 12
thyself 12
make 11
sweet 10
shall 10
should 10
were 10
beauty's 9
eyes 9
fair 9
then 9
love 9
world 9
live 8
more 7
yourself 7
than 7
where 7
```



Ниже приведен фрагмент кода с использованием функции strtok() для разбиения строки на подстроки, разделенные определенными символами (strtok() работает с `char*`, а не `string`):

```
// этот фрагмент кода должен находиться во внешнем цикле считывания строк из файла

const size_t MAXLEN = 1000;

char text[MAXLEN];
f.getline(text, MAXLEN);

char* substr = std::strtok(text, ".,:!;? ");

while (substr != 0)
{
	string word = substr;

	// преобразование всех символов строки к нижнему регистру
	std::transform(word.begin(), word.end(), word.begin(), ::tolower);

	// ... здесь необходимая работа со словом word ...

	substr = std::strtok(NULL, ".,:!;? ");
}
```

Задание 3.

Основной код функции `main()` приведен ниже. Вам необходимо написать класс `Book` с геттерами и два функтора `BookSorter` и `BookFinder`, а также уметь объяснить весь код.

```
setlocale(LC_ALL, "RUSSIAN");

std::vector<Book*> books;

books.push_back(new Book("Война и мир", "Толстой Л.Н.", 2010));
books.push_back(new Book("Подросток", "Достоевский Ф.М.", 2004));
books.push_back(new Book("Обрыв", "Гончаров И.А.", 2010));
books.push_back(new Book("Анна Каренина", "Толстой Л.Н.", 1999));
books.push_back(new Book("Обыкновенная история", "Гончаров И.А.", 2011));
books.push_back(new Book("Утраченные иллюзии", "Бальзак О.", 2009));
books.push_back(new Book("Оливер Твист", "Диккенс Ч.", 2001));
books.push_back(new Book("Фауст", "Гёте И.В.", 2010));
books.push_back(new Book("Лилия долины", "Бальзак О.", 1998));


std::cout << "\nКниги в алфавитном порядке:\n\n";

BookSorter book_sorter;
std::sort(books.begin(), books.end(), book_sorter);

std::vector<Book*>::iterator i;
for (i = books.begin(); i != books.end(); ++i)
{
    std::cout << (*i)->getAuthor() << " \""
              << (*i)->getName() << "\"" << std::endl;
}



BookFinder book_finder(2005, 2014);
std::vector<Book*>::iterator finder = std::find_if(books.begin(), books.end(), book_finder);

std::cout << "\nКниги в диапазоне года издания 2005 - 2014:\n\n";

while (finder != books.end())
{
    std:cout << (*finder)->getAuthor() << " \""
             << (*finder)->getName() << "\"" << std::endl;

    finder = std::find_if(++finder, books.end(), book_finder);
}


for (i = books.begin(); i != books.end(); ++i)
{
    delete (*i);
}
```

Задание 5.
Пример использования класса `Cache<T>`:

```
int main()
{
    Cache<int> cache;
    cache.put(1);               // так должно работать
    cache.put(2);
    cache.put(3);
    cache += 5;                 // так тоже должно работать

    Cache<std::string> voc;
    voc.put("OK");

    std::cout << voc.contains("Only") << std::endl;  // 1
    std::cout << cache.contains(5) << std::endl;     // 1

    return 0;
}
```