Информация для размещения на официальном сайте ГБПОУ «Светлоградский региональный сельскохозяйственный колледж»

Для электронного обучения

Группа	421
Дата	20.11.2021 г
Время	10-10-11-00
Наименование УД/МДК/УП/ПП	МДК 01.02
Ф.И.О. преподавателя	Сахарчук Т.В.
Электронная почта	saharchyk777@mail.ru
Основная литература	1. Интеллектуальные системы и технологии. Учебник и практикум для СПО Станкевич Л. А. Научная школа: Санкт-Петербургский политехнический университет Петра Великого (г. Санкт-Петербург). Год: 2019 / Гриф УМО СПО https://biblio-online.ru/book/intellektualnye-sistemy-i-tehnologii-445852 2. Федорова Г.Н. Разработка и администрирование баз данных (2-е изд., стер.) учебник«Академия»2019 г. 3. Федорова Г.Н. Информационные системы (6-е изд., стер.) учебник «Академия» 2019г 4. Рудаков А.В. Технология разработки программных продуктов (12-е изд.) учебник«Академия»2018 г.
Тема № 69-70	Лекция на тему: Потоки. Манипуляторы и форматирование ввода-вывода.
Задание	Обмен данными между программой и внешними устройствами осуществляется с помощью операций ввода-вывода. Типичным внешним устройством является терминал. На терминале можно напечатать информацию. Можно ввести информацию с терминала, напечатав ее на клавиатуре. Другим типичным устройством является жесткий или гибкий диск, на котором расположены *файлы*. Программа может создавать *файлы*, в которых хранится информация. Другая (или эта же) программа может читать информацию из *файла*. В языке Си++ нет особых операторов для ввода или вывода данных. Вместо этого имеется набор классов, стандартно поставляемых вместе с компилятором, которые и реализуют основные операции ввода-вывода. Причиной является как слишком большое разнообразие операций ввода и вывода в разных операционных системах, особенно графических, так и возможность определения новых типов данных в языке Си++. Вывод даже простой строки текста в MS DOS, MS Windows и в X Window настолько различен, что пытаться придумать общие для всех них операторы было бы слишком негибко и на самом деле затруднило бы работу. Что же говорить о классах, определенных программистом, у которых могут быть совершенно специфические требования к их вводу-выводу. Библиотека классов для ввода-вывода решает две задачи. Во-первых, она обеспечивает эффективный ввод-вывод всех встроенных типов и простое, но тем не менее гибкое, определение операций ввода-вывода для новых типов, разрабатываемых

программистом. Во-вторых, сама библиотека позволяет при необходимости развивать её и модифицировать.

В нашу задачу не входит описание программирования в графических системах типа MS Windows. Мы будем рассматривать операции ввода-вывода *файлов* и алфавитно-цифровой вывод на терминал, который будет работать на консольном окне MS Windows, MS DOS или Unix.

Потоки

Механизм для ввода-вывода в Си++ называется ***потоком*** . Название произошло от того,что информация вводится и выводится в виде *потока* байтов – символ за символом.

Класс istream реализует *поток* ввода, класс ostream – *поток* вывода. Эти классы определены в *файле заголовков* iostream.h. Библиотека *потоков* ввода-вывода определяет три глобальных объекта: cout, cin и cerr. cout называется стандартным выводом, cin – стандартным вводом, cerr – стандартным *потоком* сообщений об ошибках. cout и cerr выводят на терминал и принадлежат к классу ostream, cin имеет тип istream и вводит с терминала. Разница между cout и cerr существенна в Unix – они используют разные дескрипторы для вывода. В других системах они существуют больше для совместимости.

Вывод осуществляется с помощью операции <<, ввод с помощью операции >>. Выражение

```
cout << "Пример вывода: " << 34;
```

напечатает на терминале строку "Пример вывода", за которым будет выведено число 34. Выражение

```
int x;
cin >> x;
```

введет целое число с терминала в переменную x. (Разумеется, для того, чтобы ввод произошел, на терминале нужно напечатать какое-либо число и нажать клавишу возврат каретки.)

Операции << и >> для потоков

В классах iostream операции >> и << определены для всех встроенных типов языка Си++ и для строк (тип char*). Если мы хотим использовать такую же запись для ввода и вывода других классов, определенных в программе, для них нужно определить эти операции.

```
class String
{
public:
  friend  ostream& operator<<(ostream& os,
                        const String& s);
  friend  istream& operator>>(istream& is,
                        String& s);
private:
    char*  str;
    int  length;
};
ostream& operator<<(ostream& os,
                const String& s)
```

	{
	os << s.str;
	return os;
	}
	istream& operator>>(istream& is,
	String& s)
	{
	// предполагается, что строк длиной более
	// 1024 байтов не будет
	char tmp[1024];
	is >> tmp;
	
	if (s.str != 0) {
	delete [] s.str;
	}
	size_t length = strlen(tmp);
	s.str = new (std::nothrow) char[length + 1];
	if (s.str == 0) {
	// обработка ошибок
	return is;
	}
	strcpy(s.str, tmp);
	return is;
	}
	Как показано в примере класса String, операция <<, во-первых, является не методом класса String, а отдельной функцией. Она и не может быть методом класса String, поскольку ее левый операнд – объект класса ostream. С точки зрения записи, она могла бы быть методом класса ostream, но тогда с добавлением нового класса приходилось бы модифицировать класс ostream, что невозможно – каждый бы модифицировал стандартные классы, поставляемые вместе с компилятором. Когда же операция <<реализована как отдельная функция, достаточно в каждом новом классе определить ее, и можно использовать запись:
	String x;
	. . .
	cout << "this is a string: " << x;
	Во-вторых, операция << возвращает в качестве результата ссылку на *поток* вывода. Это позволяет использовать ее в выражениях типа приведенного выше, соединяющих несколько операций вывода в одно выражение.
	Аналогично реализована операция ввода. Для класса istream она определена для всех встроенных типов языка Си++ и указателей на строку символов. Если необходимо, чтобы класс, определенный в программе, позволял ввод из *потока*, для него нужно определить операцию >> в качестве функции friend.

Контрольный тест	Ответьте на вопросы: 1. С помощью чего осуществляется обмен данными между программой и внешними устройствами? 2. Охарактеризуйте внешние устройства

	3. Какие задачи решает библиотека классов? 4. Что называется потоком? 5. Охарактеризуйте операции << и >> для потоков. 6. Что такое манипуляторы? 7. Охарактеризуйте форматирование ввода-вывода.

Дата____20.11.2021 г _____ _____ _____

Подпись Ф.И.О. преподавателя

Информация для размещения на официальном сайте ГБПОУ «Светлоградский региональный сельскохозяйственный колледж»

Для электронного обучения

Группа	421
Дата	20.11.2021 г
Время	11-10-12-00
Наименование УД/МДК/УП/ПП	МДК 01.02
Ф.И.О. преподавателя	Сахарчук Т.В.
Электронная почта	saharchyk777@mail.ru
Основная литература	1. Интеллектуальные системы и технологии. Учебник и практикум для СПО Станкевич Л. А. Научная школа: Санкт-Петербургский политехнический университет Петра Великого (г. Санкт-Петербург). Год: 2019 / Гриф УМО СПО https://biblio-online.ru/book/intellektualnye-sistemy-i-tehnologii-445852 2. Федорова Г.Н. Разработка и администрирование баз данных (2-е изд., стер.) учебник«Академия»2019 г. 3. Федорова Г.Н. Информационные системы (6-е изд., стер.) учебник «Академия» 2019г 4. Рудаков А.В. Технология разработки программных продуктов (12-е изд.) учебник«Академия»2018 г.
Тема № 71-72	Лекция на тему: Строковые потоки. Ввод-вывод файлов

Задание	**Строковые потоки**
	Специальным случаем *потоков* являются *строковые потоки*, представленные классом strstream. Отличие этих *потоков* состоит в том, что все операции происходят в памяти. Фактически такие *потоки* формируют форматированную строку символов, заканчивающуюся нулевым байтом. *Строковые потоки* применяются, прежде всего, для того, чтобы облегчить форматирование данных в памяти.
	Например, в приведенном в предыдущей главе классе Exception для исключительной ситуации можно добавить сообщение. Если мы хотим составить сообщение из нескольких частей, то может возникнуть необходимость форматирования этого сообщения:
	// произошла ошибка
	strstream ss;
	ss << "Ошибка ввода-вывода, регистр: "
	 << oct << reg1;
	ss << "Системная ошибка номер: " << dec
	 << errno << ends;
	String msg(ss.str());
	ss.rdbuf()->freeze(0);
	Exception ex(Exception::INTERNAL_ERROR, msg);
	throw ex;
	Сначала создается объект типа strstream с именем ss. Затем в созданный *строковый поток* выводятся сформатированные нужным образом данные. Отметим, что в конце мы вывели *манипулятор* ends, который добавил необходимый для символьной строки байтов нулевой байт.

	Метод str() класса strstream предоставляет доступ к сформатированной строке (тип его возвращаемого значения – char*). Следующая строка освобождает память, занимаемую *строковым потоком* (подробнее об этом рассказано ниже). Последние две строки создают объект типа Exception с типом ошибки INTERNAL_ERROR и сформированным сообщением и вызывают исключительную ситуацию.
	Важное свойство класса strstream состоит в том, что он автоматически выделяет нужное количество памяти для хранения строк. В следующем примере функция split_numbers выделяет числа из строки, состоящей из нескольких чисел, разделенных пробелом, и печатает их по одному на строке.
	`#include <strstream.h>`
	`void`
	`split_numbers(const char* s)`
	`{`
	`strstream iostr;`
	`iostr << s << ends;`
	`int x;`
	`while (iostr >> x)`
	`cout << x<< endl;`
	`}`
	`int`
	`main()`
	`{`
	`split_numbers("123 34 56 932");`
	`return 1;`
	`}`
	Замечание. В среде Visual C++ файл заголовков называется strstream.h.
	Как видно из этого примера, независимо от того, какова на самом деле длина входной строки, объект iostr автоматически выделяет память, и при выходе из функции split_numbers, когда объект уничтожается, память будет освобождена.
	Однако из данного правила есть одно исключение. Если программа обращается непосредственно к хранимой в объекте строке с помощью метода str (), то объект перестает контролировать эту память, а это означает, что при уничтожении объекта память не будет освобождена. Для того чтобы память все-таки была освобождена, необходимо вызвать метод rdbuf()->freeze(0)

Контрольный тест	Ответьте на вопросы: 1. Как представлены строковые потоки? 2. Что формируют строковые потоки? 3. Какой объект создается вначале? 4. С помощью каких функций выполняется ввод-вывод файлов? 5. Как происходит запись в файл? 6. Как можно создать поток вывода с помощью стандартного конструктора без аргументов?

Дата____20.11.2021 г _____ _____ _____

Подпись Ф.И.О. преподавателя