برمجة :C ++من تحليل المشكلات إلى تصميم البرنامجو الطبعه الخامسة

C++ Programming: From Problem Analysis to Program Design, Fifth Edition

Chapter 8: The class string & File input output Handling

الفصل :8سلسلة الصف & إخراج إدخال الملف معالجة

`string` Type

- To use the data type `string`, the program must include the header file `string`
- The statement:

```
string name = "William Jacob";
```

declares name to be a string variable and also initializes name to `"William Jacob"`

  - The first character, `'W'`, is in position 0
  - The second character, `'i'`, is in position 1
  - `name` is capable of storing any size string

- لاستخدام نوع البيانات خيط، يجب أن يتضمن البرنامج ملف الرأس خيط
- البيان :

اسم السلسلة " =ويليام جاكوب "؛

يعلن أن الاسم متغير سلسلة ويهيئ أيضًا الاسم إلى "وليام جاكوب"

  - الشخصية الأولى ، "W"، في الموضع 0
  - الشخصية الثانية ، 'أنا'، في الموضع 1
  - اسم قادر على تخزين أي حجم سلسلة

خيط اكتب)تابع(

string Type (cont'd.)

- Binary operator + and the array subscript operator [], have been defined for the data type string
  - + performs the string concatenation operation
- Example:

```
str1 = "Sunny";

str2 =  str1 + " Day";
```

stores "Sunny Day" into str2

- عامل ثنائي +وعامل خط المصفوفة [] ، تم تعريفها لنوع البيانات خيط
  - +ينفذ عملية سلسلة السلسلة
- مثال :

```
" = str1  مشمس"؛

" + str2 = str1 يوم "؛
```

المخازن "يوم مشمس "إلى str2

Expression	Effect
`strVar.at(index)`	Returns the element at the position specified by `index`.
`strVar[index]`	Returns the element at the position specified by `index`.
`strVar.append(n, ch)`	Appends `n` copies of `ch` to `strVar`, in which `ch` is a `char` variable or a `char` constant.
`strVar.append(str)`	Appends `str` to `strVar`.
`strVar.clear()`	Deletes all the characters in `strVar`.
`strVar.compare(str)`	Compares `strVar` and `str`. (This operation is discussed in Chapter 4.)
`strVar.empty()`	Returns `true` if `strVar` is empty; otherwise, it returns `false`.
`strVar.erase()`	Deletes all the characters in `strVar`.
`strVar.erase(pos, n)`	Deletes `n` characters from `strVar` starting at position `pos`.
`strVar.find(str)`	Returns the index of the first occurrence of `str` in `strVar`. If `str` is not found, the special value `string::npos` is returned.

`strVar.find(str, pos)`	Returns the index of the first occurrence at or after `pos` where `str` is found in `strVar`.
`strVar.find_first_of (str, pos)`	Returns the index of the first occurrence of any character of `strVar` in `str`. The search starts at `pos`.
`strVar.find_first_not_of (str, pos)`	Returns the index of the first occurrence of any character of `str` not in `strVar`. The search starts at `pos`.
`strVar.insert(pos, n, ch);`	Inserts `n` occurrences of the character `ch` at index `pos` into `strVar`; `pos` and `n` are of type `string::size_type`; `ch` is a character.
`strVar.insert(pos, str);`	Inserts all the characters of `str` at index `pos` into `strVar`.
`strVar.length()`	Returns a value of type `string::size_type` giving the number of characters `strVar`.

Additional `string` Operations

`string::size_type`	An unsigned integer (data) type
`string::npos`	The maximum value of the (data) type `string::size_type`, a number such as `4294967295` on many machines

لمحة عن الطبقات

سلاسل كفئة C++ محددة مسبقًا

A Glimpse at Classes

Strings as a pre-defined C++ Class

```
class classIdentifier
{
    classMembersList
};
```

الطبقات

Classes

- فصل :جمع عدد ثابت من المكونات)الأعضاء(
- صيغة التعريف:

- Class: collection of a fixed number of components (members)
- Definition syntax:

  - يحدد نوع البيانات ، لا يتم تخصيص ذاكرة
  - لا تنس الفاصلة المنقوطة بعد قوس الإغلاق

  - Defines a data type, no memory is allocated
  - Don’t forget the semicolon after closing brace

Classes (cont'd.)

- Class member can be a variable or a function
- If a member of a `class` is a variable
  - It is declared like any other variable
- In the definition of the `class`
  - You cannot initialize a variable when you declare it
- If a member of a `class` is a function
  - Function prototype is listed
  - Function members can (directly) access any member of the `class`

- يمكن أن يكون عضو الفصل متغيرًا أو دالة
- إذا كان عضوا في صف دراسي متغير
  - يتم الإعلان عنه مثل أي متغير آخر
- في تعريف صف دراسي
  - لا يمكنك تهيئة متغير عندما تعلنه
- إذا كان عضوا في صف دراسي هي وظيفة
  - تم سرد النموذج الأولي للوظيفة
  - يمكن لأعضاء الوظيفة)الوصول إلى أي عضو في صف دراسي(مباشرة

Classes (cont'd.)

- Three categories of class members
  - `private` (default)
    - Member cannot be accessed outside the `class`
  - `public`
    - Member is accessible outside the class
  - `protected`

- ثلاث فئات من أعضاء الصف
  - نشر)إفتراضي(
    - لا يمكن الوصول إلى العضو خارج صف دراسي
  - عامة
    - العضو يمكن الوصول إليه من خارج الفصل
  - محمي

Classes (cont'd.)

```
class clockType
{
public:
    void setTime(int, int, int);
    void getTime(int&, int&, int&) const;
    void printTime() const;
    void incrementSeconds();
    void incrementMinutes();
    void incrementHours();
    bool equalTime(const clockType&) const;

private:
    int hr;
    int min;
    int sec;
};
```

إعلان متغير)كائن(

Variable (Object) Declaration

- Once a `class` is defined, you can declare variables of that type

```
clockType      myClock;
clockType      yourClock;
```

- A `class` variable is called a `class` object or `class` instance

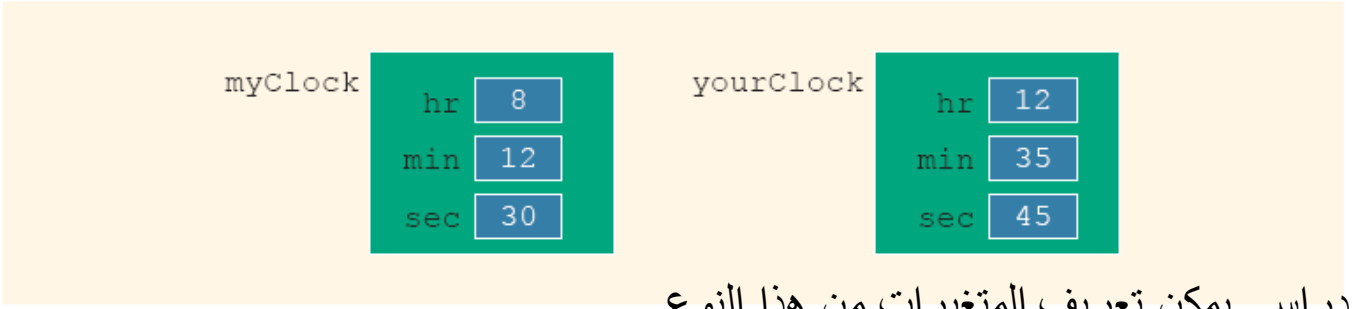

FIGURE 12-2 Objects `myClock` and `yourClock`

- مرة صف دراسي يمكن تعريف المتغيرات من هذا النوع

```
ClockType myClock ؛
ClockType yourClock ؛
```

- أ صف دراسي المتغير يسمى أ صف دراسي كائن أو صف دراسي نموذج

```
classObjectName.memberName
```

الوصول إلى أعضاء الفصل الدراسي

Accessing Class Members

- بمجرد الإعلان عن كائن ، يمكنه الوصول إلى ملف عامة أعضاء الفصل
- بناء الجملة:

- Once an object is declared, it can access the `public` members of the class
- Syntax:

  – The dot (`.`) is the ***member access operator***

- If object is declared in the definition of a member function of the `class`, it can access the `public` and `private` members

  – النقطة (.) هل ***عضو وصول عامل***

- إذا تم التصريح عن الكائن في تعريف وظيفة عضو في صف دراسي، يمكنه الوصول إلى عامة و نشر أفراد

رجوع الى *فئة السلسلة*
فئة محددة مسبقًا في C ++

Back to the *String Class*
A pre-defined Class in C++

Example 8-14

clear, *empty*, *erase*, *length*, & *size* FUNCTIONS

```
string firstName = "Elizabeth";
string name = firstName + " Jones";
string str1 = "It is sunny.";
string str2 = "";
string str3 = "computer science";
string str4 = "C++ programming.";
string str5 = firstName + " is taking " + str4;

string::size_type len;
```

Example 8-14 P463

Next, we show the effect of `clear`, `empty`, `erase`, `length`, and `size` functions.

Statement	Effect
`str3.clear();`	`str3 = "";`
`str1.empty();`	Returns `false`;
`str2.empty();`	Returns `true`;
`str4.erase(11, 4);`	`str4 = "C++ program.";`
`cout << firstName.length() << endl;`	Outputs 9
`cout << name.length() << endl;`	Outputs 15
`cout << str1.length() << endl;`	Outputs 12
`cout << str5.size() << endl;`	Outputs 36
`len = name.length();`	The value of `len` is 15

Example 8-15

find FUNCTION

```
str1.find(str2)
str1.find("the")
str1.find('a')
str1.find(str2 + "xyz")
str1.find(str2 + 'b')
```

Example 8-15
P465

Consider the following statements:

```
string sentence = "Outside it is cloudy and warm.";
string str = "cloudy";

string::size_type position;
```

Next, we show the effect of the `find` function.

Statement	Effect
`cout << sentence.find("is") << endl;`	Outputs 11
`cout << sentence.find('s') << endl;`	Outputs 3
`cout << sentence.find(str) << endl;`	Outputs 14
`cout << sentence.find("the") << endl;`	Outputs the value of `string::npos`
`cout << sentence.find('i', 6) << endl;`	Outputs 8
`position = sentence.find("warm");`	Assigns 25 to `position`

Example 8-16

insert & *replace*

FUNCTIONS

Example 8-16 P467

```
string firstString = "Cloudy and warm.";
string secondString ="Hello there";
string thirdString = "Henry is taking programming I.";
string str1 = " very ";
string str2 = "Lisa";
```

Next, we show the effect of `insert` and `replace` functions.

Statement	Effect
`firstString.insert(10, str1);`	`firstString = "Cloudy and very warm."`
`secondString.insert(11, 5, '!');`	`secondString = "Hello there!!!!!"`
`thirdString.replace(0, 5, str2);`	`thirdString = "Lisa is taking programming I."`

Example 8-17

substr FUNCTION

```
string sentence;
string str;

sentence = "It is cloudy and warm.";
```

Example 8-17
P468

Next, we show the effect of the `substr` function.

Statement	Effect
`cout << sentence.substr(0, 5) << endl;`	Outputs: `It is`
`cout << sentence.substr(6, 6) << endl;`	Outputs: `cloudy`
`cout << sentence.substr(6, 16) << endl;`	Outputs: `cloudy and warm.`
`cout << sentence.substr(17, 10) << endl;`	Outputs: `warm.`
`cout << sentence.substr(3, 6) << endl;`	Outputs: `is clo`
`str = sentence.substr(0, 8);`	`str = "It is cl"`
`str = sentence.substr(2, 10);`	`str = " is cloudy"`

تحدي البرمجة

Programming Challenge

Write a program that read your full name (first and last) in one string using *getline()* function, and then prints back your last name.

اكتب برنامجًا يقرأ اسمك بالكامل)الأول والأخير (في سلسلة واحدة باستخدام *الحصول على خط ()*وظيفة ، ثم يقوم بطباعة اسمك الأخير.

```
#include <iostream>
#include<string>
using namespace std;

int main()
{
int Pos, lnamePos, lnameLength;
string name;
string lname;

cout<<"Enter you first name and last name: ";
getline(cin, name);
```

```
Pos=name.find(' ');
lnamePos= Pos+1;
lnameLength=name.length()-lnamePos;

lname=name.substr(lnamePos, lnameLength);

cout<<"\n Your last name is: "<<lname<<endl<<endl<<endl;

system("PAUSE");

return 0;
}
```

File input output handling

A quick Overview

معالجة إخراج إدخال الملف
نظرة عامة سريعة

- File: area in secondary storage to hold info
- File I/O is a five-step process
  1. Include `fstream` header
  2. Declare file stream variables
  3. Associate the file stream variables with the input/output sources
  4. Use the file stream variables with >>, <<, or other input/output functions
  5. Close the files

- ملف :منطقة في التخزين الثانوي لعقد المعلومات
- ملف الإدخال /الإخراج هو عملية من خمس خطوات
  1. يشمل `fstream`رأس
  2. إعلان متغيرات دفق الملف
  3. اربط متغيرات تدفق الملفات بمصادر الإدخال /الإخراج
  4. استخدم متغيرات تدفق الملفات مع <<و >>، أو وظائف الإدخال /الإخراج الأخرى
  5. أغلق الملفات

مثال البرمجة
P163 - نص

Programming Example

P163 – in Text

Write a program that reads a student name followed by five test scores from a file. The program should output the student name, the five test scores, and the average test score to a file. Output the average test score with two decimal places. The data to be read is stored in a file called *test.txt*. The output should be stored in a file called *testavg.out*.

Input a file, *test.txt*, containing the student name and the five test scores. A sample input is:

Andrew Miller 87.50 89 65.75 37 98.50

Output The student name, the five test scores, and the average of the five test scores, saved to a file, *testavg.txt*.

اكتب برنامجًا يقرأ اسم الطالب متبوعًا بخمس درجات اختبار من ملف .يجب أن يُخرج البرنامج اسم الطالب ودرجات الاختبارات الخمسة ومتوسط درجات الاختبار إلى ملف .أخرج متوسط درجات الاختبار بمنزلتين عشريتين .يتم تخزين البيانات المراد قراءتها في ملف يسمى *test.txt*. يجب تخزين الإخراج في ملف يسمى *testavg.out*.

إدخال ملف، *test.txt*، التي تحتوي على اسم الطالب ودرجات الاختبارات الخمسة .إدخال العينة هو:

أندرو ميلر 87.50 89 65.75 37 98.50

انتاج | اسم الطالب ودرجات الاختبارات الخمسة ومتوسط درجات الاختبارات الخمسة المحفوظة في ملف ، *testavg.txt*.

```
//*************************************************************
// Author: D.S. Malik - Chapter 3, page 163
//
// Program to calculate the average test score.
// Given a student's name and five test scores, this program
// calculates the average test score. The student's name, the
// five test scores, and the average test score are stored in
// the file testavg.out. The data is input from the file
// test.txt.
//*************************************************************
#include <iostream>
#include <fstream>
#include <iomanip>
#include <string>
using namespace std;

main()
{
//Declare variables; Step 1
ifstream inFile;
ofstream outFile;
double test1, test2, test3, test4, test5;
double average;
```

```
string firstName;
string lastName;

inFile.open("test.txt"); //Step 2
outFile.open("testavg.out"); //Step 3
outFile << fixed << showpoint; //Step 4
outFile << setprecision(2); //Step 4
cout << "Processing data" << endl;
inFile >> firstName >> lastName; //Step 5
outFile << "Student name: " << firstName<< " " << lastName << endl; //Step 6
inFile >> test1 >> test2 >> test3>> test4 >> test5; //Step 7
outFile << "Test scores: " << setw(6) << test1<< setw(6) << test2 << setw(6) << test3<< setw(6) << test4
<< setw(6) << test5<< endl; //Step 8

average = (test1 + test2 + test3 + test4+ test5) / 5.0; //Step 9

outFile << "Average test score: " << setw(6)<< average << endl; //Step 10

inFile.close(); //Step 11
outFile.close(); //Step 11

return 0;
}
```

Sample Run:

Input File (contents of the file test.txt):
Andrew Miller 87.50 89 65.75 37 98.50

Output File (contents of the file testavg.out):
Student name: Andrew Miller
Test scores: 87.50 89.00 65.75 37.00 98.50
Average test score: 75.55