**LESSON SET 12**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Files**

**OBJECTIVES FOR STUDENT**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Lesson 12A:**

1. To review the basic concept of files

2. To use files as parameters

3. To understand and use get and ignore functions

4. To understand and work with binary files and the write function

**Lesson 12B:**

5. To understand and use random access files

6. To understand and use the tellg and seekg functions

**ASSUMPTIONS**

**Lesson 12A:**

1. Students understand and can do formatted output.

2. Students understand basic I/O (cin and cout) statements.

3. Students understand the get function and can pass files as parameters.

4. Students have read about binary files and the write function.

**Lesson 12B:**

1. Students have read about random access files.
2. Students have read about the tellg and seekg functions.

**PRE-LAB WRITING ASSIGNMENT SOLUTIONS**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. text

2. seekg

3. fstream

4. tellp

5. binary

6. seekp

7. write

8. skips

9. peek

10. get getline

**LAB ASSIGNMENTS**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Lesson 12A:**

Lab 12.1: Introduction to files (Optional)

Lab 12.2: Files as parameters and character data

Lab12.3: Binary files and the write function

**Lesson 12B:**

Lab 12.4: Random Access Files

Lab 12.5: Student generated code assignments

**LESSON 12A**

**LAB 12.1: Introduction to Files (Optional)**

Lab 12.1 gives students practice working with basic input and output file manipulation. Since this material is given early in the course, instructors may skip this lab or assign it as a review of basic file manipulation.

The program files.cpp is very similar to the samples given in the pre-lab reading material. This is a simple program that determines gross and net pay based on hours worked, pay rate and both state and federal tax rates. Students fill in code to define and open a file, and “prime the read”. Exercise 3 asks students to develop a data file and Exercise 5 has the output from the program sent to an output file. The expected output from Exercise 4 is as follows:

Payrate Hours Grosspay Netpay

15.00 40.00 600.00 498.00

10.00 50.00 500.00 420.00

12.50 60.00 750.00 615.00

A solved program is found in filesKey.cpp in the instructor’s folder for Lesson Set

12.

**LAB 12.2: Files as Parameters and Character Data**

Lab 12.2 gives students practice with both files as parameters and the use of the function

get to input character data. Program Grades.cpp asks students to develop

the prototype of the readIt function that has three parameters of the following

data types: ifstream, gradeType (array of records of struct Grades) and int.

Students may have some problem recognizing gradeType as the data type for the

array of records. The students must fill in code to bring in data (use of the get

function) in the readIt function. Expected output from Exercise 1 is as follows:

Dean DeFino 88 98 99 A

Sally Johnson 78 89 82 B

Bill Benny 75 79 81 C

Thomas Billows 78 84 89 B

Exercise 2 gives a review of structured data types.

A solved program is found in GradesKey.cpp in the instructor’s folder for

Lesson Set 12.

**LAB 12.3: Binary Files and the write Function**

This lab tests the students’ knowledge of binary files and the write and read functions.

This program reads from the keyboard the following fields of a record:

name, income, rent, food, utilities, and miscell. The program computes the net

(income minus expenses) and stores this in the net field. The entire record is then

written to a binary file (indata). This file is then closed and reopened as an input

file and read back into the record. The information of that record is then stored

in an output file. Although writing to a binary file and then closing and reopening it as an

input file may not be done often in practical situations, the exercise gives students

practice in working with binary files. Exercise 2 asks the students to expand the

program to include more than one record.

A solved program for Exercise 1 is found in budgetKey1.cpp and a solved program for Exercise 2 is found in budgetKey2.cpp in the instructor’s folder for Lesson Set 12.

**LESSON 12B**

**Lab 12.4: Random Access Files**

This lab tests the students’ ability to work with random access files and the use of the tellg and seekg functions. Exercise 1 asks the students to fill in the code that will both show and set the current read position in the file. Exercise 2 asks why the last run of the sample input displayed the previous character even though it appears that the position was changed. During the last run the offset went beyond the end of the file marker and thus we have incorrect information. The user has to close and reopen the file to have the marker start at the beginning of the file.

A solved program for Exercise 1 is found in randomAccessKey.cpp in the instructor’s folder for Lesson Set 12.

Exercise 3 asks the student to alter the program so that the read position is calculated from the end of the file. The user has to enter negative offsets to get characters from the proverb.

A solved program for Exercise 3 is found in randomAccessKey2.cpp in the instructor’s folder for Lesson Set 12.

**LAB 12.5: Student Generated Code Assignments**

Since this material is challenging, all three options are similar to the program in

12.3. All three options have the student read an array of records from the keyboard

or file, store this information into a binary file, close the binary file and

reopen it as an input file so that that information is read back into the array and

then stored in an output text file.

*Option 1:* This program will read an array of records of address information

from a keyboard and store it in a binary file. The binary file is then closed

and reopened as an input file to have the information stored back in the

array of records and then printed to an output file (with headings).

A solved program is found in addressKey.cpp in the instructor’s folder for

Lesson Set 12.

*Option 2:* This program reads the radii of circles. It is an array of records

where the radius of each circle is read from the keyboard. The program

finds the area and circumference and writes this information to a binary

file. The binary file is then closed and reopened as an input file to have

the information stored back in the array of records and then printed to an

output file (with headings). Students should include the cmath library.

A solved program is found in circleKey.cpp in the instructor’s folder for Lesson Set 12.

*Option 3:* This program is similar to the other two options except it uses an

existing data file. The students are asked to write a program that will read

records from this file and store them in a binary file. The binary file is then

closed and reopened as an input file to have the information stored back

in the array of records and then printed to an output file (with headings).

The data file contains employee information consisting of name, social

security, department id, and years employed. In addition to displaying the

information of each record the program will calculate the average salary

and years employed of all the records and have this information stored in

the output file.

A solved program is found in employeestatKey.cpp in the instructor’s folder for Lesson Set 12.