KAUNAS UNIVERSITY OF TECHNOLOGY

Faculty of Mathematics and Natural Sciences

Fundamentals of Object Programming (P175B013)

Project Report

Prepared by:

student Stephen Oluwabukunmi Adesina,

group MGMFNA-1

Supervisor:

Liudas Motiejūnas

KAUNAS

CONTENT

1. Data Grouping

1.1. Problem

Pr2-4. Book

- Student rents several books in the library. Create a class Book to store the variables for the author, the number of pages, and price. The data are given in a text file. The first line holds the name and surname of the student. The other lines contain given information for the books. Find which book is the cheapest and how much all books cost.

- Extend the program to calculate for two separate students. The data of the students are kept in separate files. Find which student rented the cheapest book. Create a new collection to store the books from both lists, which have a number of pages lesser than the cheapest book has.

1.2. Source Code

book.h

```
#pragma once
#include <string>

class Book{
//private variables for internal calculations
	private:
		std::string author;
		int no_of_pages;
		double price;
//public methods that can be accessed from outside the class to access the
//private variables.
	public:
		void setAuthor(std::string _author);
		void setPrice(double _price);
		void setNoPages(int nop);

		std::string getAuthor();
		int getNoPages();
		double getPrice();

};
```

book.cpp

```
#include "book.h"

//function to set author provided externally to the private variable
void Book::setAuthor(std::string _author){  author = _author; }
```

```
//function to set no of pages provided externally to the private variable
void Book::setNoPages(int nop){ no_of_pages = nop; }

//function to set price provided externally to the private variable
void Book::setPrice(double _price){ price = _price; }

//function to get private variables externally
std::string Book::getAuthor(){ return author;  }
int Book::getNoPages(){return no_of_pages;}
double Book::getPrice(){ return price; }
```

main.cpp

```
#include <stdlib.h>
#include <limits>
#include <sstream>
#include <iomanip>
#include <iostream>
#include <string>
#include "book.h"
#include <fstream>
#include <stdio.h>
#include <algorithm>

//a struct used to return multiple values on a number of the functions.
struct CheapReturn{
	std::string author;
	double cheapest;
	int no_of_pages;
};

//prototypes for the function defined beneath the main
void ReadData(std::string file, std::string &student_name,Book books[], int &nc);
void WriteMainData(std::string file, std::string student_name,Book books[], int nc,bool
student);
void WriteCalcData(std::string file, std::string result);
CheapReturn CheapestBook(Book books[], std::string student_name,int nc,bool write);
void SumAllBooks(Book books[], std::string student_name,int nc);
CheapReturn CheapestRent(Book books[],Book books_1[],std::string student_name,
std::string student_name_1,int nc, int nc_1,bool write);
void PagesLessThanCheapest(Book books[], Book books_1[], std::string student_name,
std::string student_name_1,int nc, int nc_1);
int PageLessAux(int pages_cheap,int nc,int count,Book books[], Book ReturnBook[]);
```

```
int main(){
	//state variables needed for various parts of the program
	const int arr_size =100;
	Book books[arr_size],books_1[arr_size];
	std::string file;
	int nc,nc_1; std::string student_name, student_name_1;

	//empty result file before reading and writing
	std::ofstream("Result.txt",std::ofstream::trunc);

	//reading data
	ReadData("Data.txt",student_name,books,nc);
	ReadData("Data2.txt",student_name_1,books_1,nc_1);

	//write data
	WriteMainData("Result.txt",student_name,books,nc,true);
	WriteMainData("Result.txt",student_name_1,books_1,nc_1,true);


	//cheapest book and sum of all books for first student
	CheapestBook(books,student_name,nc,true);
	SumAllBooks(books, student_name,nc);

	//cheapest book and sum of all books for second student
	CheapestBook(books_1,student_name_1,nc_1,true);
	SumAllBooks(books_1, student_name_1,nc_1);

	//Cheapest book rented
	CheapestRent(books,books_1,student_name,student_name_1,nc,nc_1,true);

	//Create Collection of books with less pages that the cheapest book
	PagesLessThanCheapest(books, books_1,student_name,student_name_1,nc,nc_1);
	return 0;
}

//the main function calculate which books have a number of pages less than the cheapest
book
void PagesLessThanCheapest(Book books[], Book books_1[], std::string student_name,
std::string student_name_1,int nc,int nc_1){
	Book ReturnBook[nc+nc_1];
int pages_cheap =
CheapestRent(books,books_1,student_name,student_name_1,nc,nc_1,false).no_of_pages;
//std::string author;
int count_0 = PageLessAux(pages_cheap,nc,-1,books, ReturnBook);
int count = PageLessAux(pages_cheap,nc_1,count_0,books_1, ReturnBook);
	if (count>0){
	WriteMainData("Result.txt",student_name,ReturnBook,count+1,false);
	}
	else{
		WriteCalcData("Result.txt","The cheapest book has the least number of
pages");
	}
```

```
}

//a helper function calculate which books have a number of pages less than the cheapest
book
int PageLessAux(int pages_cheap,int nc,int count,Book books[], Book ReturnBook[]){

std::string author;
int no_of_pages;
double price;
for(int i=0;i<nc;i++){
	int index = count + 1;
	if(books[i].getNoPages()<pages_cheap){
		author = books[i].getAuthor();
		ReturnBook[index].setAuthor(author);
		price = books[i].getPrice();
		ReturnBook[index].setPrice(price);
		no_of_pages = books[i].getNoPages();
		ReturnBook[index].setNoPages(no_of_pages);
		count += 1;
	}
}
return count;
}

//function to calculate the cheapest book borrowed considering all books borrowd by all
students
CheapReturn CheapestRent(Book books[],Book books_1[],std::string student_name,
std::string student_name_1,int nc, int nc_1, bool write){
	double cheapest = 0.0;std::string result;
	int no_of_pages;
	CheapReturn cheap_1 = CheapestBook(books,student_name,nc,false);
	CheapReturn cheap_2 = CheapestBook(books_1,student_name_1,nc_1,false);
 if (cheap_1.cheapest<cheap_2.cheapest){
	no_of_pages = cheap_1.no_of_pages;
  result = "\nThe cheapest was book borrowed by " + student_name+" written by " +
cheap_1.author + " and costs " + std::to_string(cheap_1.cheapest) +"\n";
 }
 else if(cheap_1.cheapest==cheap_2.cheapest){
	no_of_pages = std::min(cheap_1.no_of_pages,cheap_2.no_of_pages);
 result = "\nBoth students borrowed a book that was of the same price which was the
cheapest \nStudent 1: \n Name: " + student_name+ "\nAuthor: " + cheap_1.author + "\n
Cost: "+std::to_string(cheap_1.cheapest) +"\nStudent 2: \n Name: " + student_name+
"\nAuthor: " + cheap_2.author + "\n Cost: "+std::to_string(cheap_2.cheapest) + "\n";
 }
 else{
 no_of_pages = cheap_2.no_of_pages;
 result = "\nThe cheapest book was borrowed by " + student_name_1+" written by " +
cheap_2.author + " and costs " + std::to_string(cheap_2.cheapest) +"\n";
}

   if (write){WriteCalcData("Result.txt",result);}
  return{"",0.0,no_of_pages};
}
```

```
//function that checks the cheapest book borrowed by an individual student
CheapReturn CheapestBook(Book books[], std::string student_name,int nc, bool write){
	double cheapest = std::numeric_limits<double>::max();
	bool more=false;int no_of_pages;
	std::string name, result;
	for(int i=0;i<nc;i++){
		if(books[i].getPrice() < cheapest){
		cheapest = books[i].getPrice();
		name = books[i].getAuthor();
		no_of_pages = books[i].getNoPages();
		}
		else if(books[i].getPrice() == cheapest){
			more = true;
			cheapest = books[i].getPrice();
			name += " ," +books[i].getAuthor()+", ";
			no_of_pages = books[i].getNoPages();
		}

	}

	if (write){

		if (more){

				result = "\n More than one book has the same cheapest price.
The authors are: " + name + "and each costs " + std::to_string(cheapest)+"\n";
		}
		else{
		result = "\nThe cheapest book borrowed by " + student_name+" was written by
" + name + " and costs " + std::to_string(cheapest) +"\n";
		}
		WriteCalcData("Result.txt",result);
	}
				return {name,cheapest,no_of_pages};


}

//function to calculate the sum of all books borrowed by a student
void SumAllBooks(Book books[], std::string student_name,int nc){
	int sum_ = 0;
	for(int i=0;i<nc;i++){
	sum_ += books[i].getPrice();
	}
	std::string result = "\nThe sum of all prices for books borrowed by " +
student_name + " is " + std::to_string(sum_) + "\n";
	WriteCalcData("Result.txt",result);
}

//function to write data calculated later that might have a different
//format from original data to be written
void WriteCalcData(std::string file, std::string result){
	std::ofstream fd(file, std::ios::app);
	fd << std::setprecision(2)<<result;
```

```
	fd.close();
}

//function to read data from the file
void ReadData(std::string file,std::string &student_name, Book books[], int &nc){
	std::ifstream fd(file);
	std::string first_name, last_name;int nop;double _price;
fd >> nc; fd.ignore();
fd>> first_name; fd>> last_name; fd >> std::ws;
student_name = first_name + " " + last_name;
for (int i=0; i<nc; i++){

	fd>> first_name; fd>> last_name; fd >> std::ws; last_name.pop_back();
	books[i].setAuthor(first_name + " " + last_name);
	fd >> nop;if (nop>0) {books[i].setNoPages(nop);}
else{WriteCalcData("Result.txt","Your number of pages is negative");std::exit(0);}
	fd >> _price; if (_price>0) {books[i].setPrice(_price);}
else{WriteCalcData("Result.txt","Your price cannot be negative");std::exit(0);}
	fd.ignore();
}
fd.close();
}


//function to write data to file
void WriteMainData(std::string file, std::string student_name,Book books[], int nc,bool
student){
	std::ofstream ft(file,std::ios::app);
ft.setf(std::ios::fixed); ft.setf(std::ios::left);
ft << "\nNumber of books: " << nc << std::endl;
if(student){
ft << "Name of Student: " << student_name << std::endl;
}
ft << "List of Books: \n";
ft << "-----------------------------------------\n";
ft << "|     Author   |   No of Pages  |   Price    | \n";
ft << "-----------------------------------------\n";
	for (int i=0; i<nc; i++){
ft << "|   " << std::setw(15) << books[i].getAuthor() << " |   " << std::setprecision(1)
<< std::setw(5) << books[i].getNoPages() << "   |   "  <<std::setw(6)<<
books[i].getPrice() << " |   "  << std::endl;
	}
ft << "-----------------------------------------\n"<<std::endl;
	ft.close();
}
```

1.3. *Initial Data and Results*

Data 1.txt

```
5
Stephen Adesina
Stephen Hawking, 77 125
Dan Brown, 250 100.00
Arome Osayi, 1 1
Myles Munroe, 200 50
Wole Soyinka, 150 100
```

Data 2.txt

```
5
Schonberger Jillian
Stephen Hawking, 79 123
Marcus Brown, 210 120.00
Arome Osayi, 150 50
Musa Munroe, 180 180
Wole Genius, 50 150
```

Result.txt

```
Number of books: 5
Name of Student: Stephen Adesina
List of Books:
---------------------------------------------
|     Author    |   No of Pages  |   Price |
---------------------------------------------
|   Stephen Hawking |   77     |   125.0  |
|   Dan Brown      |   250     |   100.0  |
|   Arome Osayi    |   1       |   1.0     |
|   Myles Munroe   |   200     |   50.0   |
|   Wole Soyinka   |   150     |   100.0  |
---------------------------------------------


Number of books: 5
Name of Student: Schonberger Jillian
List of Books:
---------------------------------------------
|     Author    |   No of Pages  |   Price |
---------------------------------------------
|   Stephen Hawking |   79     |   123.0  |
|   Marcus Brown   |   210     |   120.0  |
|   Arome Osayi    |   150     |   50.0   |
```

```
|   Musa Munroe   |   180     |   180.0  |
|   Wole Genius   |   50      |   150.0  |
-------------------------------------------


The cheapest book borrowed by Stephen Adesina was written by Arome Osayi and
costs 1.000000

The sum of all prices for books borrowed by Stephen Adesina is 376

The cheapest book borrowed by Schonberger Jillian was written by Arome Osayi and
costs 50.000000

The sum of all prices for books borrowed by Schonberger Jillian is 623

The cheapest was book borrowed by Stephen Adesina written by Arome Osayi and
costs 1.000000
The cheapest book has the least number of pages
```