**INSTALL 4 U**

**FURNITURE INSTALLATION BOOKING PROGRAM**

**FINAL PROJECT**

**MIRNA E. SUMOPAWIRO**

**2001586344**

**COMP6336: INTRODUCTION TO PROGRAMMING**

Table of Contents

SECTION I: DESCRIPTION3

SECTION II: DESIGN3

PLAN3

DETAILS OF PLAN3

**SECTION III: REFLECTION6**

KNOWLEDGE GAINED6

PROBLEMS ENCOUNTERED6

**SECTION IV: CODE7**

**SECTION IV: BIBLIOGRAPHY14**

**INSTALL 4 U: FURNITURE INSTALLATION BOOKING PROGRAM**

**SECTION I: DESCRIPTION**

With the development of technology, people all over the world can now order various items online. From food to taxi rides, everything is just a click away. However, it is still very uncommon for people to book an installer to come and install certain furniture. Many would still have to go through the hardships of finding an installer, calling the installer and then setting up a date. If the installer could not install their furniture, they would have to find another installer.

This is where Install 4 U comes in. Install 4 U is a program designed to allow users to book an installation for furniture. The purpose of this program is to ease the booking process by merely typing the furniture choice, amount, and customer details instead of having to call an installer or come to the store directly. This program uses four functions and two classes. Knowledge applied in this program includes if statements, switch, loop, functions, classes and exceptions.

This program is inspired by Install 2 U Assembly & Installation Services, Australia.

**SECTION II: DESIGN**

**PLAN**

This program will use four functions, namely the Main function, the Show Menu function, the Get Data function and the Show Summary function. This program will also use two classes, which are Orders and CustomerData.

The hierarchy chart of this program is as follow:

**DETAILS OF PLAN**

Details of Functions:

1. Main

This is the main function of the program. Here, the functions Show Menu, Get Data and Show Summary will be called. Also, objects from Orders and CustomerData classes will be made. An exception thrown by the Show Menu function will be caught here as well, if any. Validations on whether the customer would like to proceed to the next step or to cancel the booking will also be found in the main function.

1. Show Menu

This function will display a menu where the user will be able to choose which furniture would like to be installed. The menu also aids the user in inputting orders by showing how to enter the order. In this case, the user should only enter the number on the left hand side of the furniture instead of typing down the name of the furniture itself. Choices 1-5 are furniture models and choice 6 is for the user to quit the program. If the user input any number other than 1-6, the program will ask the user to enter a number between 1-6. If the user enters 6, the program will throw an exception string, which will be caught in the main function and the program will be aborted. The furniture choice and its amount that was entered in this function will then be put into an object made from class Orders.

1. Get Data

The purpose of this function is to accept the user’s data, namely the user’s first name, phone number and city. These will be the details that the installer needs from the customer to do the installation. The idea is that the installer would call the customer after the booking is done, which is why the precise address and installation date is not necessary. Data’s entered in this function will then be put into an object made from class CustomerData.

1. Show Summary

This function displays the summary of the booking. The summary includes the furniture choice, the amount of that furniture to be installed, the customer’s personal data (first name, phone number, city) and also the quotation details (subtotal, call out fee, and grand total). This is where the customer makes sure that the booking details are correct.

Details of Classes:

1. Orders

This class will hold the order details of the installation. Its members are divided into private and public access specifications.

Members that are in private are the prices of each furniture’s in float data type, the choice and also the amount of the furniture that the user input in integer data type, and also the subtotal and grand total of the booking in float data type.

Constructors, accessor and mutator functions are specified as public-accessed in this class. There is one constructor with two parameters, five accessor functions, to get the furniture choice, amount, subtotal, call out fee and grand total, and also two mutator functions, to set the subtotal and grand total. The accessor for furniture choice and amount has integer return type and the rest has float return type. All mutator functions have void return type.

1. CustomerData

This class will hold the customer’s details. The details include the customer’s first name, number and city. Since the customer will be called when the booking is done, the customer should only enter their first name. These details are specified as private.

A constructor and three accessor functions share public access specification. The constructor has three parameters, which are the name, number and city, and the accessor functions gets the customer’s name, number and city respectively.

**SECTION III: REFLECTION**

**KNOWLEDGE GAINED**

Many new things have been learned from conducting this project. Although the main focus is on practicing the application of the theories, skills in solving several errors have been acquired as well. I have learned how to decide whether to use a structure or a class and I have also learned how to make my program more efficient.

In addition to those mentioned above, I have also familiarized myself to recheck every line of my program to check for minor and major errors such as syntax error and logical errors. From this conducting this project, I have gained an understanding of how important it is to label functions or variables with comments to make the code more readable.

**PROBLEMS ENCOUNTERED**

In the process of making this program, numerous difficulties were faced. Syntax errors were faced the most and it is not rare for me to easily panic when my program does not run when I try to compile it. It is often for me to change a part of the header of a class but forgetting to change that part when declaring its function. For instance, I changed the return type from float to void in a header file. When I try to compile it, it would fail because I forgot to change the return type when declaring its function in the .cpp file.

Other problems I encountered were when using a structure, which I initially planned to use. I had problems when attempting to use an array. After learning about classes, I realized that it would be a wiser choice to use class instead of a structure and so that is what I did.

Time management is crucial in the process of making this program and I have failed to manage my time wisely in the earlier stages. It has been a struggle trying to balance adapting to university courses as well as maintaining social life whilst trying to make sure I get enough sleep and meals. However, I did manage to complete this project on time.

After facing all of these problems, I have learned many things that will be important for me in the years ahead. I am very thankful to the lecturers who contributed in this course and have designed this project since it has been a roller coaster of learning experiences.

**SECTION IV: CODE**

Contents of Orders.hpp

#ifndef Orders\_hpp

#define Orders\_hpp

#include <iostream>

using namespace std;

//Orders Class

class Orders

{

private:

const float BOOKCASE\_PRICE = 40.91; //Price of a bookcase

const float BED\_WITH\_STORAGE\_PRICE = 86.36; //Price of a bed with under storage

const float BED\_WO\_STORAGE\_PRICE = 40.91; //Price of a bed without under storage

const float TV\_BENCH\_PRICE = 86.36; //Price of a TV Bench

const float WARDROBE\_PRICE = 172.72; //Price of a wardrobe

const float CALL\_OUT\_FEE = 31.82; //Call out fee. A mandatory fee for the installer to come

int furnitureChoice;

int furnitureAmount;

float subTotal; //furniture amount \* price of furniture

float grandTotal; //subtotal + call out fee

public:

//Constructor

Orders (int furnitureChoice, int furnitureAmount);

//Accessors and mutators

int getChoice();

int getAmount();

void setSubTotal(int furnitureChoice, int furnitureAmount);

float getSubTotal();

float getCallOutFee();

void setGrandTotal();

float getGrandTotal();

};

#endif /\* Orders\_hpp \*/

Contents of Orders.cpp

#include "Orders.hpp"

Orders::Orders (int furnitureChoice, int furnitureAmount)

{

this->furnitureChoice = furnitureChoice;

this->furnitureAmount = furnitureAmount;

}

int Orders::getChoice()

{

return furnitureChoice;

}

int Orders::getAmount()

{

return furnitureAmount;

}

void Orders::setSubTotal(int furnitureChoice, int furnitureAmount)

{

switch (furnitureChoice)

{

case 1:

subTotal = furnitureAmount \* BOOKCASE\_PRICE;

break;

case 2:

subTotal = furnitureAmount \* BED\_WITH\_STORAGE\_PRICE;

break;

case 3:

subTotal = furnitureAmount \* BED\_WO\_STORAGE\_PRICE;

break;

case 4:

subTotal = furnitureAmount \* TV\_BENCH\_PRICE;

break;

case 5:

subTotal = furnitureAmount \* WARDROBE\_PRICE;

}

}

float Orders::getSubTotal()

{

return subTotal;

}

float Orders::getCallOutFee()

{

return CALL\_OUT\_FEE;

}

void Orders::setGrandTotal()

{

grandTotal = subTotal + CALL\_OUT\_FEE;

}

float Orders::getGrandTotal()

{

return grandTotal;

}

Contents of CustomerData.hpp

#ifndef CustomerData\_hpp

#define CustomerData\_hpp

#include <iostream>

using namespace std;

//CustomerData class

class CustomerData

{

private:

string name;

string number;

string city;

public:

//Constructor

CustomerData (string name, string number, string city);

//Accessors

string getName();

string getNumber();

string getCity();

};

#endif /\* CustomerData\_hpp \*/

Contents of CustomerData.cpp

#include "CustomerData.hpp"

CustomerData::CustomerData (string name, string number, string city)

{

this->name = name;

this->number = number;

this->city = city;

}

string CustomerData::getName()

{

return name;

}

string CustomerData::getNumber()

{

return number;

}

string CustomerData::getCity()

{

return city;

}

Contents of main.cpp

#include <iostream>

#include <iomanip>

#include "Orders.hpp"

#include "CustomerData.hpp"

using namespace std;

int fChoice; //input furniture choice

int fAmount; //input furniture amount

string inName; //input name

string inNumber; //input phone number

string inCity; //input city

char choice; //input choice

//function prototypes

void showMenu();

void getData();

void showSummary();

int main()

{

cout << "\t\t\t WELCOME TO INSTALL 4 U\n";

try

{

showMenu(); //calling Show Menu function to display menu and accept orders

}

catch (string exceptionString)

{

cout << exceptionString;

exit (0);

}

Orders orderInput (fChoice, fAmount); //creating an object from Orders class

cout << "\t\t\tORDER SUMMARY:" << endl;

cout << "Furniture number: " << setw(2) << fixed << orderInput.getChoice() << endl;

cout << "Amount to install: " << setw(2) << fixed << orderInput.getAmount() << endl;

orderInput.setSubTotal(fChoice, fAmount);

cout << endl;

cout << setprecision(2);

cout << "Subtotal: " << setw(2) << fixed << "$" << orderInput.getSubTotal() << endl;

cout << "Would you like to proceed? (Y/N)\n";

cin >> choice;

if (choice == 'Y' || choice == 'y')

{

getData(); //calls Get Data function to accept customer's details

}

else

{

cout << "Thank you for visiting our website" << endl;

cout << "We look forward to your next booking" << endl;

exit(0);

}

CustomerData customerDetails (inName, inNumber, inCity); //creating an object from CustomerData class

showSummary(); //displays the summary of the order

cout << "Would you like to book your installation now?" << endl;

cin >> choice;

if (choice == 'Y' || choice == 'y')

{

cout << "Thank you for booking an installation with us" << endl;

cout << "Our installer will call you within 48 hours. Have a nice day!" << endl;

}

else

{

cout << "Thank you for visiting our website" << endl;

cout << "We look forward to your next booking" << endl;

exit(0);

}

return 0;

}

//End of program

//FUNCTION DEFINITIONS

//1. Show Menu function

void showMenu()

{

cout << "Which furniture would you like us to install for you? (ex: for Bookcase, enter 1)\n";

cout << "1. Bookcase \n";

cout << "2. Bed with Under Storage \n";

cout << "3. Bed without Under Storage \n";

cout << "4. TV Bench \n";

cout << "5. 2-frame Wardrobe with Hinged Doors \n";

cout << "6. I do not want to book an installation right now\n";

cout << "Order: ";

cin >> fChoice;

if (fChoice < 1 || fChoice > 6)

{

cout << "Please enter a number between 1-6" << endl;

cout << "Order: ";

cin >> fChoice;

}

if (fChoice == 6)

{

string exceptionString = "Thank you for visiting our website\nWe look forward to your next booking\n";

throw exceptionString;

}

cout << "How much of this would you like to install?\n";

cin >> fAmount;

}

//2. Get Data function

void getData ()

{

do

{

cout << "Please enter the following details\n";

cout << "First name: ";

cin >> inName;

cout << "Phone number: ";

cin >> inNumber;

cout << "City: ";

cin >> inCity;

cout << endl;

cout << "CONFIRM DETAILS" << endl;

cout << "Name: " << inName << endl;

cout << "Phone number: " << inNumber << endl;

cout << "City: " << inCity << endl;

cout << "Would you like to change your details? (Y/N)";

cin >> choice;

}

while (choice == 'Y' || choice == 'y');

}

//3. Show Summary function

void showSummary()

{

CustomerData customerDetails (inName, inNumber, inCity);

Orders orderInput (fChoice, fAmount);

cout << endl;

cout << "\t\t\tBOOKING SUMMARY" << endl;

cout << endl;

cout << endl;

cout << "CUSTOMER DETAILS" << endl;

cout << "Name: " << "\t\t\t" << customerDetails.getName() << endl;

cout << "Phone number: " << "\t" << customerDetails.getNumber() << endl;

cout << "City: " << "\t\t\t" << customerDetails.getCity() << endl;

cout << endl;

cout << "ORDER" << endl;

cout << "Furniture number: " << "\t\t" << orderInput.getChoice() << endl;

cout << "Amount to install: " << "\t\t" << orderInput.getAmount() << endl;

cout << endl;

cout << "QUOTATION" << endl;

orderInput.setSubTotal(fChoice, fAmount);

cout << setprecision(2);

cout << "Subtotal: " << "\t\t" << "$" << orderInput.getSubTotal() << endl;

cout << "Call Out Fee:" << "\t\t" << "$" << orderInput.getCallOutFee() << endl;

orderInput.setGrandTotal();

cout << "Grand Total:" << "\t\t" "$" << orderInput.getGrandTotal() << endl;

}

**SECTION VI: BIBLIOGRAPHY**

Gaddis, T. (2015). *Starting out with C From Control Structures through Objects* (8th ed.). Boston: Pearson Addison-Wesley.

Install2u - IKEA Flat Pack Furniture Assembly & Installation Services. (n.d.). Retrieved November 06, 2016, from http://install2u.com.au/