Case Study: High Adventure Travel Agency—Part 3

In Case Study 5, a modification to the High Adventure Travel Agency program was started, which will be completed here.

Recall that in Case Study 5, data structures were designed to hold all the cost information for a vacation package. You are now ready to add file I/O capabilities. When this section is complete, the program will be able to save the cost information of booked vacation packages to a reservation file. It will also display a list of all the data stored in the file. The menu that is displayed lists the four vacation packages, plus a fifth option that causes the data stored in the reservation file to be displayed. A sixth option exits the program.

New Functions

Three new functions, will be added to the program: `openFile`, `saveInfo`, and `showRes`. Table 1 describes the purpose of each function.

Table 1 New Functions

Function Name	Description
`openFile`	Opens the requested file.
`saveInfo`	Saves the cost data of a vacation package reservation to the file currently open.
`showRes`	Displays all of the package reservation data in the file currently open.

The `openFile` Function

The `OpenFile` function is called prior to the menu being displayed. It first asks the user to enter the name of a file. That file is then opened for input and output, in binary mode. Here is the pseudocode:

```
openFile Function
    Ask user for file name.
    Open file in binary mode for input and output.
    If file failed to open
        Create the file.
    End If.
End Function.
```

The C++ code is shown here:

```
void openFile(fstream &file)
{
    const int SIZE = 256;
    char fileName[SIZE];

    cout << "File name: ";
    cin.getline(fileName, SIZE);

    file.open(fileName, ios::in | ios::out | ios::binary);
    if (!file)
    {
        cout << "Creating " << fileName << "...\n";
        // Create the file.
        file.open(fileName, ios::out);
        // Close the file.
        file.close();
        // Reopen the file for input and output.
        file.open(fileName, ios::in | ios::out | ios::binary);
    }
}
```

The `saveInfo` Function

The `saveInfo` function is called after the costs for a specific vacation package have been calculated. The user is asked if he or she wants to save the data. If so, the data structure holding the data is written to the reservation file currently open. Here is the pseudocode:

```
saveInfo Function
    Ask user "Do you want to save this data?".
    If "yes"
        Write record to the file.
        If write operation failed
            Display error message.
        End If.
    End If.
End Function.
```

The C++ code is shown here:

```
void saveInfo(Reservation &group, fstream &file)
{
    char yorN;

    cout << "Do you want to save this data? (Y/N) ";
    cin >> yorN;
    yorN = toupper(yorN);
```

```
        // Validate input
        while (yorN != 'Y' && yorN != 'N')
        {
            cout << "Please enter Y or N\n";
            cin >> yorN;
            yorN = toupper(yorN);
        }

        // Save the data.
        if (yorN == 'Y')
        {
            cout << "Saving reservation data.\n";
            file.write(reinterpret_cast<char *>(&group), sizeof(group));
            if (!file)
                cout << "Could not write to file!\n";
        }
    }
```

The showRes Function

When the user selects option 5 from the menu, the `showRes` function is called. This function moves the file's read position to the beginning of the file. It then begins a loop that displays the record's data, and reads the next record from the file. The loop repeats until the end of the file is encountered. Here is the pseudocode:

```
showRes Function
      Seek beginning of file.
      Read a record from the file.
      While not at the end of the file
          Display the record.
          Ask user to press a key to continue.
          Read a record from the file.
      End While.
End Function.
```

The C++ code is shown here:

```
void showRes(fstream &file)
{
    Reservation temp;
    char skip[2];

    file.seekg(0L, ios::beg);  // Go to beginning of file.
    file.read(reinterpret_cast<char *>(&temp), sizeof(temp));
    while (!file.eof())
    {
        displayInfo(temp);
        cout << "Type a character and press Enter "
             << "to continue:";
        cin >> skip;
        file.read(reinterpret_cast<char *>(&temp), sizeof(temp));
    }
    if (file.fail())
       file.clear();           // Clear any error state
}
```

The function defines a local variable, `temp`, which is a `Reservation` structure. It's used to hold each record as it is read from the file. After each record's data is displayed, the message `"Type a character and press Enter"` is displayed. When the user performs this action, the function repeats the loop, reading the next record.

The last `if` statement tests the condition of the file's `fail` bit. If `file.fail` returns true, the `fail` bit is cleared so processing may resume. (The `fail` bit may be set after the last record has been read, as a result of the read member trying to read past the end of the file.)

Program CS6-1 shows the entire program.

Program CS6-1

```
// This program will assist the High Adventure Travel Agency
// in booking reservations for any of their 4 major
// vacation packages.
#include <iostream>
#include <fstream>
#include <iomanip>
#include <cctype>
using namespace std;

// Data Structures
struct Package1                     // Climbing Package
{
    int        num;                 // Number in party
    int        beginners;           // Those needing instruction
    int        advanced;            // Those not needing instruction
    int        needEquip;           // Those renting camping equipment
    double     baseCharges;         // Base charges
    double     charges;             // Total charges
    double     instruction;         // Cost of instruction
    double     equipment;           // Cost of equipment rental
    double     discount;            // Discount
    double     deposit;             // Required deposit
};

struct Package2                     // Scuba Package
{
    int        num;                 // Number in party
    int        beginners;           // Those needing instruction
    int        advanced;            // Those not needing instruction
    double     baseCharges;         // Base charges
    double     charges;             // Total charges
    double     instruction;         // Cost of instruction
    double     discount;            // Discount
    double     deposit;             // Required deposit
};

```

```
struct Package3                  // Sky Diving Package
{
    int         num;             // Number in party
    int         lodge1;          // Number at 1st lodging choice
    int         lodge2;          // Number at 2nd lodging choice
    double      baseCharges;     // Base charges
    double      charges;         // Total charges
    double      discount;        // Discount
    double      lodging;         // Cost of lodging
    double      deposit;         // Required deposit
};

struct Package4                  // Spelunking Package
{
    int         num;             // Number in party
    int         needEquip;       // Those renting camping equipment
    double      baseCharges;     // Base charges
    double      charges;         // Total charges
    double      equipment;       // Cost of equipment rental
    double      discount;        // Discount
    double      deposit;         // Required deposit
};

union Pack
{
    struct Package1 climb;
    struct Package2 scuba;
    struct Package3 sky;
    struct Package4 spel;
};

struct Reservation
{
    int packNum;
    union Pack packs;
};

// Constants for the charges.
const double CLIMB_RATE = 350.0;             // Base rate - Devil's Courthouse
const double SCUBA_RATE = 1000.0;            // Base rate - Bahamas
const double SKY_DIVE_RATE = 400.0;          // Base rate - Sky diving
const double CAVE_RATE = 700.0;              // Base rate - Spelunking
const double CLIMB_INSTRUCT = 100.0;         // Climbing instruction
const double SCUBA_INSTRUCT = 100.0;         // Scuba instruction
const double DAILY_CAMP_RENTAL = 40.0;       // Daily camping equip. rental
const double DAY_LODGE_1 = 65.0;             // Lodging option (sky diving)
const double DAY_LODGE_2 = 120.0;            // Lodging option (sky diving)

```

(program continues)

Program CS6-1 *(continued)*

```
// Function prototypes
void openFile(fstream &);
void saveInfo(Reservation &, fstream &);
void climbing(Reservation &);
void scuba(Reservation &);
void skyDive(Reservation &);
void spelunk(Reservation &);
int menu();
void displayInfo(Reservation &);
void displayPack1(Reservation &);
void displayPack2(Reservation &);
void displayPack3(Reservation &);
void displayPack4(Reservation &);
void showRes(fstream &);

int main()
{
    int selection;
    Reservation group;
    fstream file;

    cout << fixed << showpoint << setprecision(2);
    openFile(file);
    do
    {
        selection = menu();
        switch(selection)
        {
            case 1 : climbing(group);
                     break;
            case 2 : scuba(group);
                     break;
            case 3 : skyDive(group);
                     break;
            case 4 : spelunk(group);
                     break;
            case 5 : showRes(file);
                     break;
            case 6 : cout << "Exiting program.\n\n";
        }
        if (selection < 5)
        {
            displayInfo(group);
            saveInfo(group, file);
        }
    } while (selection != 6);
    file.close();
    return 0;
}

```

```
//**************************************************
// Definition of function openFile.                *
// Accepts an fstream object as an argument. The   *
// file is opened for both input and output, in    *
// binary mode.                                    *
//**************************************************

void openFile(fstream &file)
{
    const int SIZE = 256;
    char fileName[SIZE];

    cout << "File name: ";
    cin.getline(fileName, SIZE);

    file.open(fileName, ios::in | ios::out | ios::binary);
    if (!file)
    {
        cout << "Creating " << fileName << "...\n";
        // Create the file.
        file.open(fileName, ios::out);
        // Close the file.
        file.close();
        // Reopen the file for input and output.
        file.open(fileName, ios::in | ios::out | ios::binary);
    }
}

//************************************************************
// Definition of function saveInfo.                          *
// Accepts a Reservation structure and an fstream object.    *
// The user is asked if the data in the structure            *
// is to be saved. If so, it is saved at the end of the file.*
//************************************************************

void saveInfo(Reservation &group, fstream &file)
{
    char yorN;

    cout << "Do you want to save this data? (Y/N) ";
    cin >> yorN;
    yorN = toupper(yorN);

```

(program continues)

Program CS6-1 *(continued)*

```
    // Validate input
    while (yorN != 'Y' && yorN != 'N')
    {
        cout << "Please enter Y or N\n";
        cin >> yorN;
        yorN = toupper(yorN);
    }

    // Save the data.
    if (yorN == 'Y')
    {
        cout << "Saving reservation data.\n";
        file.write(reinterpret_cast<char *>(&group), sizeof(group));
        if (!file)
            cout << "Could not write to file!\n";
    }
}

//*******************************************************
// Definition of function menu.                         *
// Displays the main menu and asks the user to select *
// an option. Returns an integer in the range 1 - 6.   *
//*******************************************************

int menu( )
{
    int choice;

    cout << "High Adventure Travel Agency\n";
    cout << "----------------------------\n";
    cout << "1) Devil's Courthouse Adventure Weekend\n";
    cout << "2) Scuba Bahama\n";
    cout << "3) Sky Dive Colorado\n";
    cout << "4) Barron Cliff Spelunk\n";
    cout << "5) Show Booked Reservations\n";
    cout << "6) Exit Program\n\n";
    cout << "Enter 1, 2, 3, 4, 5, or 6: ";
    cin >> choice;

    while (choice < 1 || choice > 6)
    {
        cout << "Invalid Selection\n";
        cin >> choice;
    }
    return choice;
}

```

```
//*********************************************************
// Definition of climbing function.                        *
// Uses a Reservation reference parameter to hold the      *
// vacation package information.                           *
// This function calculates the charges for the            *
// Devil's Courthouse Adventure Weekend package.            *
//*********************************************************

void climbing(Reservation &group)
{
    group.packNum = 1;
    cout << "\nDevil's Courthouse Adventure Weekend\n";
    cout << "------------------------------------\n";
    cout << "How many will be going who need an instructor? ";
    cin >> group.packs.climb.beginners;
    cout << "How many advanced climbers will be going? ";
    cin >> group.packs.climb.advanced;
    group.packs.climb.num = group.packs.climb.beginners +
              group.packs.climb.advanced;
    cout << "How many will rent camping equipment? ";
    cin >> group.packs.climb.needEquip;
    // Calculate base charges.
    group.packs.climb.baseCharges = group.packs.climb.num *
             CLIMB_RATE;
    group.packs.climb.charges = group.packs.climb.baseCharges;
    // Calculate 10% discount for 5 or more.
    if (group.packs.climb.num > 4)
    {
        group.packs.climb.discount = group.packs.climb.charges
                  * .1;
        group.packs.climb.charges -= group.packs.climb.discount;
    }
    else
        group.packs.climb.discount = 0;
    // Add cost of instruction.
    group.packs.climb.instruction = group.packs.climb.beginners
             * CLIMB_INSTRUCT;
    group.packs.climb.charges += group.packs.climb.instruction;
    // Add cost of camping equipment rental
    group.packs.climb.equipment = group.packs.climb.needEquip *
              DAILY_CAMP_RENTAL * 4;
    group.packs.climb.charges += group.packs.climb.equipment;
    // Calculate required deposit.
    group.packs.climb.deposit = group.packs.climb.charges / 2.0;
}

```

(program continues)

Program CS6-1 *(continued)*

```
//*******************************************************
// Definition of scuba function.                        *
// Uses a Reservation reference parameter to hold the   *
// vacation package information.                        *
// This function calculates the charges for the         *
// Scuba Bahama package.                                *
//*******************************************************

void scuba(Reservation &group)
{
    group.packNum = 2;
    cout << "\nScuba Bahama\n";
    cout << "-------------------------------------\n";
    cout << "How many will be going who need an instructor? ";
    cin >> group.packs.scuba.beginners;
    cout << "How many advanced scuba divers will be going? ";
    cin >> group.packs.scuba.advanced;
    group.packs.scuba.num = group.packs.scuba.beginners +
             group.packs.scuba.advanced;
    // Calculate base charges.
    group.packs.scuba.baseCharges = group.packs.scuba.num *
            SCUBA_RATE;
    group.packs.scuba.charges = group.packs.scuba.baseCharges;
    // Calculate 10% discount for 5 or more.
    if (group.packs.scuba.num > 4)
    {
        group.packs.scuba.discount = group.packs.scuba.charges
              * .1;
        group.packs.scuba.charges -= group.packs.scuba.discount;
    }
    else
        group.packs.scuba.discount = 0;
    // Add cost of instruction.
    group.packs.scuba.instruction = group.packs.scuba.beginners
           * SCUBA_INSTRUCT;
    group.packs.scuba.charges += group.packs.scuba.instruction;
    // Calculate required deposit.
    group.packs.scuba.deposit = group.packs.scuba.charges / 2.0;
}

//*******************************************************
// Definition of skyDive function.                      *
// Uses a Reservation reference parameter to hold the   *
// vacation package information.                        *
// This function calculates the charges for the         *
// Sky Dive Colorado package.                           *
//*******************************************************

```

```
void skyDive(Reservation &group)
{
    group.packNum = 3;
    cout << "\nSky Dive Colorado\n";
    cout << "-------------------------------------\n";
    cout << "How many will be going? ";
    cin >> group.packs.sky.num;
    // Calculate base charges.
    group.packs.sky.baseCharges = group.packs.sky.num *
                SKY_DIVE_RATE;
    group.packs.sky.charges = group.packs.sky.baseCharges;
    // Calculate 10% discount for 5 or more.
    if (group.packs.sky.num > 4)
    {
        group.packs.sky.discount = group.packs.sky.charges * .1;
        group.packs.sky.charges -= group.packs.sky.discount;
    }
    else
        group.packs.sky.discount = 0;
    // Calculate lodging costs.
    cout << "How may will stay at Wilderness Lodge? ";
    cin >> group.packs.sky.lodge1;
    cout << "How many will stay at Luxury Inn? ";
    cin >> group.packs.sky.lodge2;
    group.packs.sky.lodging = (group.packs.sky.lodge1 *
        DAY_LODGE_1) + (group.packs.sky.lodge2 * DAY_LODGE_2);
    group.packs.sky.charges += group.packs.sky.lodging;
    // Calculate required deposit.
    group.packs.sky.deposit = group.packs.sky.charges / 2.0;
}

//*********************************************************
// Definition of spelunk function.                        *
// Uses a Reservation reference parameter to hold the     *
// vacation package information.                          *
// This function calculates the charges for the           *
// Barron Cliff Spelunk package.                          *
//*********************************************************

void spelunk(Reservation &group)
{
    group.packNum = 4;
    cout << "\nBarron Cliff spelunk Weekend\n";
    cout << "-------------------------------------\n";
    cout << "How many will be going? ";
    cin >> group.packs.spel.num;
    cout << "How many will rent camping equipment? ";
    cin >> group.packs.spel.needEquip;
```

(program continues)

Program CS6-1 *(continued)*

```
     // Calculate base charges.
     group.packs.spel.baseCharges = group.packs.spel.num *
               CAVE_RATE;
     group.packs.spel.charges = group.packs.spel.baseCharges;
     // Calculate 10% discount for 5 or more.
     if (group.packs.spel.num > 4)
     {
         group.packs.spel.discount = group.packs.spel.charges * .1;
         group.packs.spel.charges -= group.packs.spel.discount;
     }
     else
         group.packs.spel.discount = 0;
     // Add cost of camping equipment rental
     group.packs.spel.equipment = group.packs.spel.needEquip *
              DAILY_CAMP_RENTAL * 4;
     group.packs.spel.charges += group.packs.spel.equipment;
     // Calculate required deposit.
     group.packs.spel.deposit = group.packs.spel.charges / 2.0;
}

//****************************************************************
// Definition of function displayInfo.                           *
// Uses a Reservation reference parameter to hold the            *
// vacation package information. This function looks in the     *
// group.packNum member to determine which function to call      *
// to display the vacation package information.                  *
//****************************************************************

void displayInfo(Reservation &group)
{
     switch (group.packNum)
     {
         case 1: displayPack1(group);
             break;
         case 2: displayPack2(group);
             break;
         case 3: displayPack3(group);
             break;
         case 4: displayPack4(group);
             break;
         default: cout << "ERROR: Invalid package number.\n";
     }
}

//****************************************************************
// Definition of function displayPack1.                          *
// Uses a Reservation reference parameter to hold the            *
// vacation package information. This function displays the      *
// information stored for vacation package 1.                    *
//****************************************************************

```

```
void displayPack1(Reservation &group)
{
    cout << "Package: Devil's Courthouse Adventure Weekend\n";
    cout << "Number in party: "
        << group.packs.climb.num << endl;
    cout << "Base charges: $"
        << group.packs.climb.baseCharges << endl;
    cout << "Instruction cost: $"
        << group.packs.climb.instruction << endl;
    cout << "Equipment rental: $"
        << group.packs.climb.equipment << endl;
    cout << "Discount: $"
        << group.packs.climb.discount << endl;
    cout << "Total charges: $"
        << group.packs.climb.charges << endl;
    cout << "Required deposit: $"
        << group.packs.climb.deposit << endl << endl;
}

//****************************************************************
// Definition of function displayPack2.                          *
// Uses a Reservation reference parameter to hold the             *
// vacation package information. This function displays the      *
// information stored for vacation package 2.                     *
//****************************************************************

void displayPack2(Reservation &group)
{
    cout << "Package: Scuba Bahama\n";
    cout << "Number in party: "
        << group.packs.scuba.num << endl;
    cout << "Base charges: $"
        << group.packs.scuba.baseCharges << endl;
    cout << "Instruction cost: $"
        << group.packs.scuba.instruction << endl;
    cout << "Discount: $"
        << group.packs.scuba.discount << endl;
    cout << "Total charges: $"
        << group.packs.scuba.charges << endl;
    cout << "Required deposit: $"
        << group.packs.scuba.deposit << endl << endl;
}

//****************************************************************
// Definition of function displayPack3.                           *
// Uses a Reservation reference parameter to hold the             *
// vacation package information. This function displays the      *
// information stored for vacation package 3.                     *
//****************************************************************

```

(program continues)

Program CS6-1 *(continued)*

```
void displayPack3(Reservation &group)
{
    cout << "Package: Sky Dive Colorado\n";
    cout << "Number in party: "
        << group.packs.sky.num << endl;
    cout << "Base charges: $"
        << group.packs.sky.baseCharges << endl;
    cout << "Lodging: $"
        << group.packs.sky.lodging << endl;
    cout << "Discount: $"
        << group.packs.sky.discount << endl;
    cout << "Total charges: $"
        << group.packs.sky.charges << endl;
    cout << "Required deposit: $"
        << group.packs.sky.deposit << endl << endl;
}

//****************************************************************
// Definition of function displayPack4.                          *
// Uses a Reservation reference parameter to hold the            *
// vacation package information. This function displays the     *
// information stored for vacation package 4.                    *
//****************************************************************

void displayPack4(Reservation &group)
{
    cout << "Package: Barron Cliff Spelunk\n";
    cout << "Number in party: "
        << group.packs.spel.num << endl;
    cout << "Base charges: $"
        << group.packs.spel.baseCharges << endl;
    cout << "Equipment rental: $"
        << group.packs.spel.equipment << endl;
    cout << "Discount: $"
        << group.packs.spel.discount << endl;
    cout << "Total charges: $"
        << group.packs.spel.charges << endl;
    cout << "Required deposit: $"
        << group.packs.spel.deposit << endl << endl;
}

//****************************************************************
// Definition of function showRes.                               *
// Accepts an fstream object as an argument. Seeks the           *
// beginning of the file and then reads and displays             *
// each record.                                                  *
//****************************************************************

```

```
void showRes(fstream &file)
{
    Reservation temp;
    char skip[2];

    file.seekg(0L, ios::beg); // Go to beginning of file.
    file.read(reinterpret_cast<char *>(&temp), sizeof(temp));
    while (!file.eof())
    {
        displayInfo(temp);
        cout << "Type a character and press Enter "
            << "to continue:";
        cin >> skip;
        file.read(reinterpret_cast<char *>(&temp), sizeof(temp));
    }
    if (file.fail())
        file.clear(); // Clear any error state
}
```

Program Output with Example Input Shown in Bold

```
File name: resfile [Enter]
High Adventure Travel Agency
----------------------------
1) Devil's Courthouse Adventure Weekend
2) Scuba Bahama
3) Sky Dive Colorado
4) Barron Cliff Spelunk
5) Show Booked Reservations
6) Exit Program

Enter 1, 2, 3, 4, 5, or 6: 1 [Enter]

Devil's Courthouse Adventure Weekend
------------------------------------
How many will be going who need an instructor? 3 [Enter]
How many advanced climbers will be going? 2 [Enter]
How many will rent camping equipment? 3 [Enter]
Package: Devil's Courthouse Adventure Weekend
Number in party: 5
Base charges: $1750.00
Instruction cost: $300.00
Equipment rental: $480.00
Discount: $175.00
Total charges: $2355.00
Required deposit: $1177.50

Do you want to save this data? (Y/N) y [Enter]
Saving reservation data.
```

(program output continues)

Program CS6-1 *(continued)*

```
High Adventure Travel Agency
----------------------------
1) Devil's Courthouse Adventure Weekend
2) Scuba Bahama
3) Sky Dive Colorado
4) Barron Cliff Spelunk
5) Show Booked Reservations
6) Exit Program

Enter 1, 2, 3, 4, 5, or 6: 3 [Enter]

Sky Dive Colorado
------------------------------------
How many will be going? 8 [Enter]
How many will stay at Wilderness Lodge? 4 [Enter]
How many will stay at Luxury Inn? 4 [Enter]
Package: Sky Dive Colorado
Number in party: 8
Base charges: $3200.00
Lodging: $740.00
Discount: $320.00
Total charges: $3620.00
Required deposit: $1810.00

Do you want to save this data? (Y/N) y [Enter]
Saving reservation data.

High Adventure Travel Agency
----------------------------
1) Devil's Courthouse Adventure Weekend
2) Scuba Bahama
3) Sky Dive Colorado
4) Barron Cliff Spelunk
5) Show Booked Reservations
6) Exit Program

Enter 1, 2, 3, 4, 5, or 6: 5 [Enter]

Package: Devil's Courthouse Adventure Weekend
Number in party: 5
Base charges: $1750.00
Instruction cost: $300.00
Equipment rental: $480.00
Discount: $175.00
Total charges: $2355.00
Required deposit: $1177.50
```

```
Type a character and press Enter to continue: g [Enter]
Package: Sky Dive Colorado
Number in party: 8
Base charges: $3200.00
Lodging: $740.00
Discount: $320.00
Total charges: $3620.00
Required deposit: $1810.00

Type a character and press Enter to continue: g [Enter]

High Adventure Travel Agency
----------------------------
1) Devil's Courthouse Adventure Weekend
2) Scuba Bahama
3) Sky Dive Colorado
4) Barron Cliff Spelunk
5) Show Booked Reservations
6) Exit Program

Enter 1, 2, 3, 4, 5, or 6: 6 [Enter]
Exiting program.
```