Experiment 7 :

Date 10-12-21

Expt. No. 7

PADRE CONCEICAO COLLEGE OF ENGINEERING
VERNA

Page No. 76

Console IO

Aim: To write C++ programs to implement the following:

1. Compare cin, cin.get() and cin.getline()
2. Demonstrate use of peek, putback and ignore.
3. Demonstrate use of read, write and gcount.
4. Print an integer value in octal, hexadecimal and decimal and setbase.
5. Demonstrate use of precision and setprecision.
6. Demonstrate use of
   a. showpoint,
   b. left, right and internal justification
   c. scientific and fixed notation
   d. boolalpha
7. program to create user defined output stream manipulators.
8. Show stream error states with examples.

Theory: Every program takes some data as input and generates processed data as an output following the familiar input process output cycle. It is essential to know how to provide the input data and present the results in the desired form.

Date

Expt. No.

PADRE CONCEICAO COLLEGE OF ENGINEERING

VERNA

Page No.

The iomanip is a library in c++ which helps us in manipulating the output of any c++ program. There are many functions in the library that help in manipulating output. It is a part of input output library of the c++ standard library. All these functions can be easily used whenever they should affect state of the iostream objects.

Some functions of the library are:

i) resetiosflags : it clears the specified ios-base flags.

ii) setiosflags : It will set the ios-base flags.

iii) setbase : It will change the base which is needed as per the said base.

iv) setfill : It will fill with the character specified.

v) setprecision : It will change the precision of any floating-point number.

vi) setw : It helps in changing the width of the next input or output field.

Programs:

1. **Compare cin, cin.get() and cin.getline()**

```
#include<iostream>
#include<conio.h>
using namespace std;

int main()
{
      const int s = 100;
      char str1[s], str2[s], str3[s];

      cout << "Enter a sentence: " << endl;
      cin >> str1;
      cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
      cout << "\nThe string read with cin was:\n" << str1 << endl;

      cout << "\nEnter a sentence: " << endl;
      cin.get(str2, 10);
      cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
    cout << "\nThe string read with cin.get was:\n" << str2 << endl;

      cout << "\nEnter a sentence: " << endl;
      cin.getline(str3, 20);
      cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
      cout << "\nThe string read with cin.getline was:\n" << str3 << endl <<
endl;

      return 0;
}
```

Output:

```
Enter a sentence:
This is a test string

The string read with cin was:
This

Enter a sentence:
This is a test string

The string read with cin.get was:
This is a

Enter a sentence:
This is a test string

The string read with cin.getline was:
This is a test stri

Press any key to continue . . .
```

2. Demostrate use of peek, putback and ignore

```
#include<iostream>
#include<conio.h>
using namespace std;
int main()
{
	char ch;
	cout << "Enter a sentence: " << endl;
	while (cin.get(ch))
	{
		if (ch == '!')
			cin.putback('$');
		else
			cout << ch;

		while (cin.peek() == '#')
			cin.ignore(1, '#');

	}
	_getch();
}
```

Output:

```
Enter a sentence:
hello!
hello$
#ello!!!
ello$$$
```

3. Demostrate use of read, write and gcount.

```
#include<iostream>
#include<conio.h>
using namespace std;
int main()
{

	const int s = 100;
	char str[s];
	cout << "Enter sentence:" << endl;
	cin.read(str, 20);
	cout << "The string read was:" << endl;
	cout.write(str,cin.gcount());
	cout << "\nThe string had " << cin.gcount() << " characters";
	_getch();
	return 0;
}
```

Output:

```
Enter sentence:
this is a test string
The string read was:
this is a test strin
The string had 20 characters
```

4. **Print an integer value in octal, hexadecimal and decimal and setbase.**

```
#include<iostream>
#include<conio.h>
#include<iomanip>
using namespace std;
int main()
{
      int n;
      cout << "Enter a decimal number" << endl;
      cin >> n;
      cout << n << " in HexaDecimal is: " << hex << n << endl;
      cout << dec << n << " in Octal is: " << oct << n << endl;
      cout << setbase(10) << n << " in decimal is: " << n << endl;
      _getch();
}
```

Output:

```
Enter a decimal number
44
44 in HexaDecimal is: 2c
44 in Octal is: 54
44 in decimal is: 44
```

5. Demostrate use of precision and setprecision

```
#include<iostream>
#include<conio.h>
#include<iomanip>
#include<cmath>
using namespace std;

int main()
{
      double r2 = sqrt(2.0);

      cout << "Square root of 2 with precision 0-9.\n";
      cout << "Precision set by ios_base member function precision: \n";
      cout << fixed;
```

```
    for (int i = 0;i <= 9;i++)
    {
    cout.precision(i);
    cout << r2 << endl;
    }
    cout << "\nPrecision set by io manipulator member function
setprecision: \n";
    for (int i = 0;i <= 9;i++)
          cout << setprecision(i) << r2 << endl;

    _getch();
    return 0;
}
```

Output:

```
Square root of 2 with precision 0-9.
Precision set by ios_base member function precision:
1
1.4
1.41
1.414
1.4142
1.41421
1.414214
1.4142136
1.41421356
1.414213562

Precision set by io manipulator member function setprecision:
1
1.4
1.41
1.414
1.4142
1.41421
1.414214
1.4142136
1.41421356
1.414213562
```

6. Demostrate use of
    a. showpoint,
    b. left, right and internal justification
    c. scientific and fixed notation
    d. boolalpha

```
#include <iostream>
#include<iomanip>
#include<conio.h>
using namespace std;
int main()
{

    cout << "Implemening showpoint:\n";
    cout << "Before using showpoint" << endl
          << "9.9900 prints as: " << 9.9900 << endl
          << "9.9000 prints as: " << 9.9000 << endl
```

```
        << "9.0000 prints as: " << 9.0000 << endl;
    cout << showpoint
        << "After using showpoint" << endl
        << "9.9900 prints as: " << 9.9900 << endl
        << "9.9000 prints as: " << 9.9000 << endl
        << "9.0000 prints as: " << 9.0000 << endl << endl;

    cout << "Implementing left, right and internal justification:\n";
    int a = 12345;
    cout << "Default is right justified:" << endl << setw(10) << a;
    cout << "\nUse std::left to left justify x:\n" << left << setw(10) <<
a;
    cout << "\nUse std::right to right justify x:\n" << right << setw(10)
<< a << endl << endl;

    cout << "Implementing scientific and fixed notation:\n";
    double x = 0.001234567;
    double y = 1.946e9;
    cout << "Displayed in default format:" << endl << x << '\t' << y <<
endl;
    cout << "Displayed in scientific format:" << endl << scientific << x <<
'\t' << y << endl;
    cout << "Displayed in fixed format:" << endl << fixed << x << '\t' << y
<< endl << endl;

    cout << "Implementing boolalpha:\n";
    bool booleanValue = true;
    cout << "BooleanValue is " << booleanValue << endl;
    cout << "BooleanValue (after using boolalpha) is " << boolalpha <<
booleanValue << endl;
    cout << "Switch booleanValue and use noboolalpha" << endl;
    booleanValue = false;
    cout << noboolalpha;
    cout << "BooleanValue is " << booleanValue << endl;
    cout << "BooleanValue (after using boolalpha) is " << boolalpha <<
booleanValue << endl;

    _getch();
    return 0;
}
```

Output:

```
Implemening showpoint:
Before using showpoint
9.9900 prints as: 9.99
9.9000 prints as: 9.9
9.0000 prints as: 9
After using showpoint
9.9900 prints as: 9.99000
9.9000 prints as: 9.90000
9.0000 prints as: 9.00000

Implementing left, right and internal justification:
Default is right justified:
    12345
Use std::left to left justify x:
12345
Use std::right to right justify x:
    12345

Implementing scientific and fixed notation:
Displayed in default format:
0.00123457      1.94600e+09
Displayed in scientific format:
1.234567e-03    1.946000e+09
Displayed in fixed format:
0.001235        1946000000.000000

Implementing boolalpha:
BooleanValue is 1
BooleanValue (after using boolalpha) is true
Switch booleanValue and use noboolalpha
BooleanValue is 0
BooleanValue (after using boolalpha) is false
```

7. program to create user defined output stream manipulators.

```
#include<iostream>
#include<conio.h>
using namespace std;
ostream& bell(ostream& output){
return output << "\a";
}
ostream& carraigeReturn(ostream& output){
return output << "\r";
}
ostream& tab(ostream& output){
return output << "\t";
}
ostream& endLine(ostream& output){
return output << "\n" << flush;
}
int main(){
     cout << "Use Of tab and endline manipulator" << endLine;
     cout << "a" << tab << "b" << tab << "c" << endLine;
     cout << "Use of carraigeReturn and bell maniplulator" << endLine;
     cout << bell;
     cout << carraigeReturn << "-----" << endLine;
     _getch();
     return 0;
}
```

Output:

```
Use Of tab and endline manipulator
a       b       c
Use of carraigeReturn and bell maniplulator
```

8. Show stream error states with examples

using namespace std;

```
int main()
{
      int integerValue;

      cout << " Before a bad input operation:"
            << "\n cin.rdstate(): " << cin.rdstate()
            << "\n cin.eof(): " << cin.eof()
            << "\n cin.fail(): " << cin.fail()
            << "\n cin.bad(): " << cin.bad()
            << "\n cin.good(): " << cin.good()
            << "\n\n Expects an integer, but enter a character: ";
      cin >> integerValue;
      cout << endl;

      cout << " After a bad input operation:"
            << "\n cin.rdstate(): " << cin.rdstate()
            << "\n cin.eof(): " << cin.eof()
            << "\n cin.fail(): " << cin.fail()
            << "\n cin.bad(): " << cin.bad()
            << "\n cin.good(): " << cin.good() << endl << endl;

      cin.clear();
      cout << " After cin.clear()" << "\n cin.fail(): " << cin.fail() << "\n
cin.good(): " << cin.good() << endl;

      _getch();
}
```

Output:

```
Before a bad input operation:
cin.rdstate(): 0
cin.eof(): 0
cin.fail(): 0
cin.bad(): 0
cin.good(): 1

Expects an integer, but enter a character: c

After a bad input operation:
cin.rdstate(): 4
cin.eof(): 0
cin.fail(): 1
cin.bad(): 0
cin.good(): 0

After cin.clear()
cin.fail(): 0
cin.good(): 1
```

PADRE CONCEICAO COLLEGE OF ENGINEERING

VERNA

11/12/2021

Conclusion: C++ programs implementing console IO were studied and executed.

Date

Expt. No. 8

PADRE CONCEICAO COLLEGE OF ENGINEERING
VERNA

Page No. 86

Templates

Aim: To write C++ programs to implement the following:

1. Bubble sort using function templates.
2. Function overloading of display(). The 3 forms will be
   a. Displaying 2 numbers of different types.
   b. Displaying 1 template type variable and 1 built-in type.
3. Program to add, subtract, multiply and divide two numbers using class template.
4. Define class Stack<> and implement generic methods to push and pop the elements from the stack.

Theory: A template is a simple and yet very powerful tool in C++. The simple idea is to pass data type as a parameter so that we dont need to write the same code for different data types. for examples, a software company may need sort() for different data types. Rather than writing and maintaining the multiple codes, we can write one sort() and pass data type as a parameter.

C++ adds two new keywords to support templates: 'template' and 'typename'. The second keyword can always be replaced by keyword 'class'.

How do templates work?
Templates are expanded at compiler time. This is like macros. The difference is, the compiler does type checking before template expansion. The idea is simple, source code only contains function/class, but compiled code may contain multiple copies of same function/class.

Function Templates: we write a generic function that can be used for different data types. Examples of function templates are sort(), max(), min(), printArray().

Class Templates: Like function templates, class templates are useful when a class defines something that is independent of the data type. Can be useful for classes like LinkedList, BinaryTree, Stack, Queue, Array, etc.

programs:

1. Bubble sort using Function templates.

```
#include <conio.h>
#include <iostream>
using namespace std;

//Declaration of template class bubble
template <class bubble>
void bubbleSort(bubble a[], int n)
{
   int i, j;

   for (i = 0; i < n - 1; i++)
   {
       for (j = i + 1; j < n; j++)
       {
           if (a[i] > a[j])
           {
               bubble b;
               b = a[i];
               a[i] = a[j];
               a[j] = b;
           }
       }
   }
}

int main()
{
   int arr[20], k, i;
   char ch[20];

       cout << "\nEnter the number of elements in integer array:";
       cin >> k;
       cout << "\nEnter elements:";
       for (i = 0; i < k; i++)
           cin >> arr[i];

       bubbleSort(arr, k);
       cout << "\nSorted integer array: ";
       for (i = 0; i < k; i++)
           cout << arr[i] << "\t";

       cout << "\nEnter the number of characters in the array:";
       cin >> k;
       cout << "\nEnter elements:";
       for (i = 0; i < k; i++)
           cin >> ch[i];

       bubbleSort(ch, k);
       cout << "\nSorted character array: ";
       for (i = 0; i < k; i++)
```

```
            cout << ch[i] << "\t";
        cout << endl;

    getch();
    return 0;
}
```

Output:

```
Enter the number of elements in integer array:5
Enter elements:45
22
42
14
2
Sorted integer array: 2 14      22      42      45
Enter the number of characters in the array:5
Enter elements:v
c
a
m
s
Sorted character array: a       c       m       s       v
```

2. Function overloading of display(). The 3 forms will be
    a. Displaying 2 numbers of different types
    b. Displaying 1 template type variable and 1 built-in type

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

template <typename T>
void print(T num1){
      cout << "Number : " << num1 << endl;
}

template <typename T>
void print(T num1, int inNum){
      cout << "Number 1: " << num1 << endl;
      cout << "Number 2: " << inNum << endl;
}

int main() {
      int inNum;
      float flNum;
      cout << "Enter a number of interger type : ";
      cin >> inNum;
      print(inNum);
      cout << "Enter a number of float type : ";
      cin >> flNum;
      print(flNum);
      cout << endl << "--- overloaded function ---" << endl;
```

```
    print(flNum, inNum);
    return 0;
}
```

Output:

```
Enter a number of interger type : 4
Number : 4
Enter a number of float type : 2.2
Number : 2.2

--- overloaded function ---
Number 1: 2.2
Number 2: 4
Press any key to continue . . .
```

3. Program to add, subtract, multiply and divide two numbers using class template.

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

template <class T>
class Calculator {
  private:
   T num1, num2;

  public:
   Calculator(T n1, T n2) {
       num1 = n1;
       num2 = n2;
   }

   void displayResult() {
       cout << "Numbers: " << num1 << " and " << num2 << endl;
       cout << num1 << " + " << num2 << " = " << add() << endl;
       cout << num1 << " - " << num2 << " = " << subtract() << endl;
       cout << num1 << " * " << num2 << " = " << multiply() << endl;
       cout << num1 << " / " << num2 << " = " << divide() << endl;
   }

   T add() { return num1 + num2; }
   T subtract() { return num1 - num2; }
   T multiply() { return num1 * num2; }
   T divide() { return num1 / num2; }
};

int main() {
   Calculator<int> intNums(40, 10);
   Calculator<float> floatNums(4.4, 2.2);

   cout << "----- Integer -----" << endl;
   intNums.displayResult();

   cout << endl << "----- Float -----" << endl;
   floatNums.displayResult();
```

```
    return 0;
}
```

Output:

```
----- Integer -----
Numbers: 40 and 10
40 + 10 = 50
40 - 10 = 30
40 * 10 = 400
40 / 10 = 4

----- Float -----
Numbers: 4.4 and 2.2
4.4 + 2.2 = 6.6
4.4 - 2.2 = 2.2
4.4 * 2.2 = 9.68
4.4 / 2.2 = 2
Press any key to continue . . .
```

4. Define class Stack<> and implement generic methods to push and pop the elements from the stack

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

#define SIZE 5

template <class T> class Stack {
public:
    Stack();
    void push(T k);
    T pop();
    T topElement();
    bool isFull();
    bool isEmpty();

private:
    int top;
    T st[SIZE];
};

template <class T> Stack<T>::Stack() { top = -1; }

template <class T> void Stack<T>::push(T k)
{


    if (isFull()) {
        cout << "Stack is full\n";
    }


    cout << "Inserted element " << k << endl;
```

```
    top += 1;
    st[top] = k;
}
template <class T> bool Stack<T>::isEmpty()
{
    if (top == -1)
        return 1;
    else
        return 0;
}
template <class T> bool Stack<T>::isFull()
{
    if (top == (SIZE - 1))
        return 1;
    else
        return 0;
}
template <class T> T Stack<T>::pop()
{
    T popped_element = st[top];
    top--;
    return popped_element;
}
template <class T> T Stack<T>::topElement()
{
    T top_element = st[top];
    return top_element;
}
int main()
{
    Stack<int> integer_stack;
    Stack<string> string_stack;

      cout << "----- Interger Stack -----" << endl;
    integer_stack.push(2);
    integer_stack.push(54);
    integer_stack.push(255);
    cout << integer_stack.pop() << " is popped from stack" << endl;
    cout << "Top element is " << integer_stack.topElement() << endl;

    cout << "\n----- String Stack -----" << endl;
    string_stack.push("Hello");
    string_stack.push("world");
    cout << string_stack.pop() << " is popped from stack " << endl;
    cout << "Top element is " << string_stack.topElement() << endl;

    return 0;
}
```

Output:

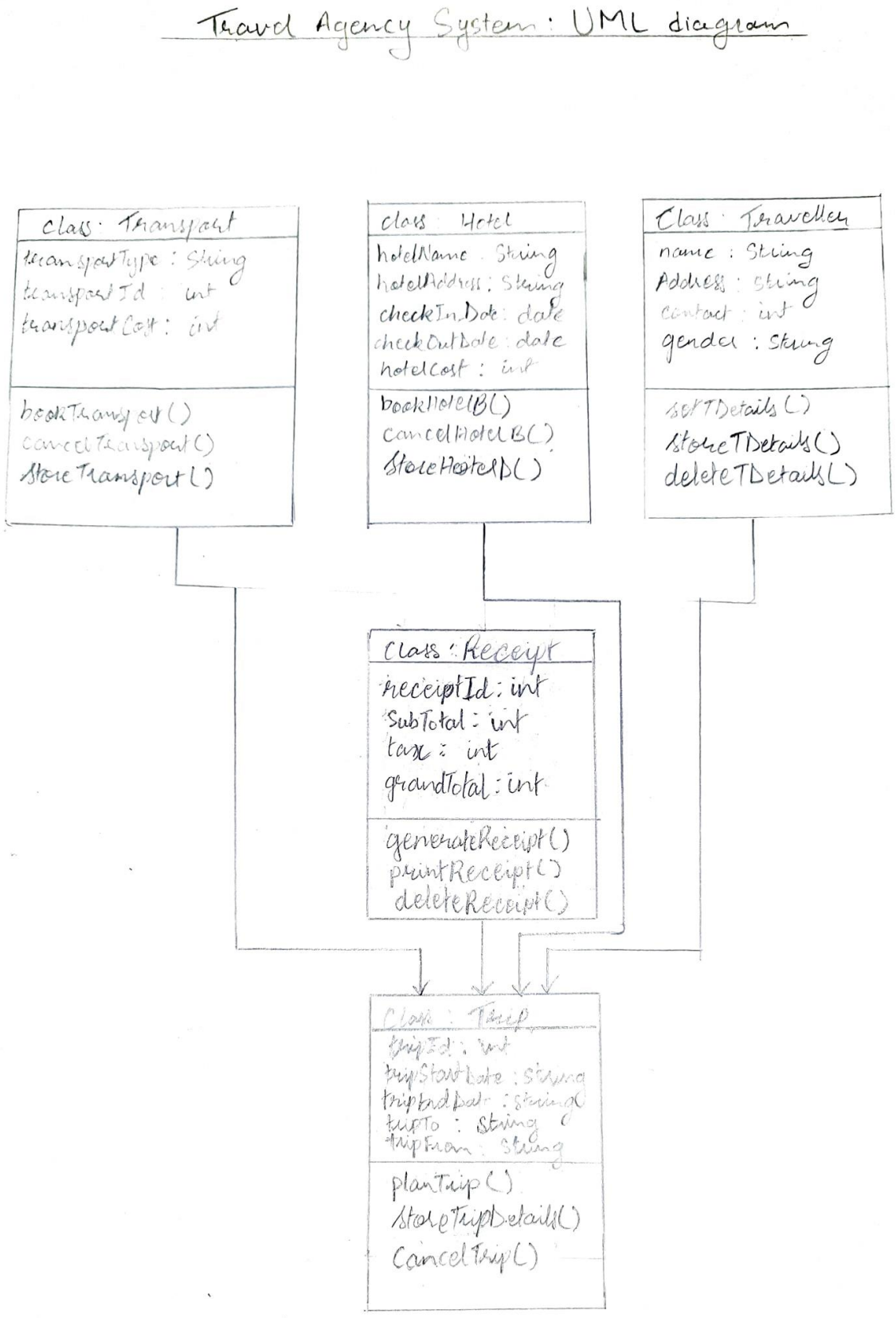
```
----- Interger Stack -----
Inserted element 2
Inserted element 54
Inserted element 255
255 is popped from stack
Top element is 54

----- String Stack -----
Inserted element Hello
Inserted element world
world is popped from stack
Top element is Hello
Press any key to continue . . .
```

PADRE CONCEICAO COLLEGE OF ENGINEERING

VERNA

Conclusion: C++ programs implementing templates were studied and executed.

Mini Project: Travel agency system



Program:

```
#include<iostream>
#include<conio.h>
#include<fstream>
#include<sstream>
#include<string>
#include<cstdlib>
#include<windows.h>
#include<iomanip>
#include<time.h>
using namespace std;

const int LINELENGTH = 80;

enum Position{ LEFT, CENTRE, RIGHT };

void clearScreen()
{
	system("cls");
	cout << "Click 'ctrl + c' to exit the program." << endl;
}

void wait(int time = 800)
{
	Sleep(time);
}

int generateRandomId(){
	wait();
	srand(time(0));
	return rand();
}

void print( Position pos, string s, int linelength )
{
   int spaces = 0;
   switch( pos )
   {
      case CENTRE: spaces = ( linelength - s.size() ) / 2; break;
      case RIGHT : spaces =   linelength - s.size()      ; break;
   }
   if ( spaces > 0 ) cout << string( spaces, ' ' );
   cout << s << '\n';
}

void print(int LINELENGTH){
	string border(LINELENGTH, '=');
	cout << border << endl;
	print( CENTRE, "Welcome To Travel Agency System", LINELENGTH );
	cout << endl << border << endl;
}

void printMainMenu(int LINELENGTH){
		print(CENTRE, "Travel Agency Menu", LINELENGTH);
	cout << "\n 1. Plan new trip." << endl;
	cout << " 2. Cancel Trip." << endl;
	cout << " 3. Exit." << endl;
	cout << "\n Choose One: ";
}

class Transport{
```

```
	protected:
	int transport_id;
	string transport_type;
	int transport_cost;

	public:
	Transport(){
		transport_id = 0;  // can have random function which will set the id
		transport_type = "";
		transport_cost = 0;
	}

	void getTransportDetails(int tripOption){
		// assigning id
		transport_id = generateRandomId();




		// taking input for transport type
		cout << "\n Enter(exact word) transport type from the following " <<
endl;
		string border(LINELENGTH,'*');
		cout << endl << border;
		cout << "|" << left << setw(20) << " " << " | " << left << setw(55) <<
"Transport type" << right << "|";
		cout << border;
		cout << "|" << left << setw(20) << " 1" << " | " << left << setw(55) <<
"Bus" << right << "|";
		cout << "|" << left << setw(20) << " 2" << " | " << left << setw(55) <<
"Flight" << right << "|";
		cout << "|" << left << setw(20) << " 3" << " | " << left << setw(55) <<
"Train" << right << "|";
		cout << "|" << left << setw(20) << " 4" << " | " << left << setw(55) <<
"Cab" << right << "|";
		cout << border << endl;
		cout << ": ";
		getline(cin, transport_type);

		// assigning the transport cost
		if(transport_type == "Bus" || transport_type == "bus" || transport_type
== "BUS"){
			transport_cost = 1200 * tripOption;
		}else if(transport_type == "Flight" || transport_type == "flight" ||
transport_type == "BUS"){
			transport_cost = 4500 * tripOption;
		}else if(transport_type == "Train" || transport_type == "train" ||
transport_type == "TRAIN"){
			transport_cost = 900 * tripOption;
		}else if(transport_type == "Cab" || transport_type == "cab" ||
transport_type == "CAB"){
			transport_cost = 4000 * tripOption;
		}else {
			transport_cost = 3000;
		}
	}

	void bookTransport(int trip_id){
		ofstream fout;
		fout.open("transport-details.txt", ios::app);
		fout << trip_id << " " << transport_id  << " "<< transport_type << " " <<
transport_cost << endl;
```

```
            fout.close();
      }

      void cancelTransport(int delete_trip_id){
            ifstream fin;
            string readLine;
            ofstream fout;

            fin.open("transport-details.txt", ios::in);
            if(fin.is_open())
            {
                  while(getline(fin, readLine))
                  {
                        int fileTripId = 0;
                        int size = readLine.length();
                        char tripD[size+1];
                        int temp = 0, i=0;
                        strcpy(tripD, readLine.c_str());
                        while((temp = tripD[i++]) != 32)
                              fileTripId = fileTripId * 10 + (temp - 48);
                        if(fileTripId != delete_trip_id)
                        {
                              fout.open("temp.txt", ios::app);
                              fout<<readLine<<endl;
                              fout.close();
                        }
                  }
                  fin.close();
            }
            remove("transport-details.txt");
            rename("temp.txt","transport-details.txt");
      }
};

class Hotel{
      protected:
      string hotel_name;
      string check_in_date;
      string check_out_date;
      int hotel_cost;

      public:
      Hotel() {
            hotel_name = "";
            check_in_date = "";
            check_out_date = "";
            hotel_cost = 0;
      }

      void printHotelMenu(int LINELENGTH){
            string border(LINELENGTH,'*');
            cout << endl << border;
            cout << "|" << left << setw(20) << "Sr No." << " | " << left << setw(30)
<< "Hotel Name" << " | " << left << setw(22) << "Price/Day" << right << "|";
            cout << border;
            cout << "|" << left << setw(20) << " 1" << " | " << left << setw(30) <<
"ITC Gardenia" << " | " << left << setw(22) << "5000" << right << "|";
            cout << "|" << left << setw(20) << " 2" << " | " << left << setw(30) <<
"Novotel" << " | " << left << setw(22) << "3000" << right << "|";
            cout << "|" << left << setw(20) << " 3" << " | " << left << setw(30) <<
"Woodville Palace" << " | " << left << setw(22) << "2500" << right << "|";
```

```
		cout << "|" << left << setw(20) << " 4" << " | " << left << setw(30) << "Ibis" << " | " << left << setw(22) << "2000" << right << "|";
		cout << border << endl;
	}

	void setHotelNameCost(int& hotelOption){
		SelectHotel:
				cout << "\n Enter Hotel sr.no: ";
				cin >> hotelOption;

	cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
				switch(hotelOption){
					case 1: hotel_name = "ITC Gardenia";
							hotel_cost = 5000;
					break;

					case 2: hotel_name = "Novotel";
							hotel_cost = 3000;
					break;

					case 3: hotel_name = "Woodville Palace";
							hotel_cost = 2500;
					break;

					case 4: hotel_name = "Ibis";
							hotel_cost = 2000;
					break;
					default: cout << "\n Enter Correct option from above..." << endl;
							goto SelectHotel;
					break;

				}
	}

	void getHotelDetails(int LINELENGTH){
		int hotelOption = 0;
		// print hotel list here
		printHotelMenu(LINELENGTH);
		setHotelNameCost(hotelOption);
		cout << " Checkin date: ";
		getline(cin, check_in_date);
		cout << " Checkout date: ";
		getline(cin, check_out_date);
	}

	void bookHotel(int trip_id){
		ofstream fout;
		fout.open("Hotel-details.txt", ios::app);
		fout << trip_id << " " << hotel_name  << " "<< check_in_date << " " << check_out_date << " " << hotel_cost << endl;
		fout.close();
	}

	void cancelHotel(int delete_trip_id){
		ifstream fin;
		string readLine;
		ofstream fout;

		fin.open("Hotel-details.txt", ios::in);
		if(fin.is_open())
		{
```

```
                        while(getline(fin, readLine))
                        {
                                int fileTripId = 0;
                                int size = readLine.length();
                                char tripD[size+1];
                                int temp = 0, i=0;
                                strcpy(tripD, readLine.c_str());
                                while((temp = tripD[i++]) != 32)
                                        fileTripId = fileTripId * 10 + (temp - 48);
                                if(fileTripId != delete_trip_id)
                                {
                                        fout.open("temp.txt", ios::app);
                                        fout<<readLine<<endl;
                                        fout.close();
                                }
                        }
                        fin.close();
                }
                remove("Hotel-details.txt");
                rename("temp.txt","Hotel-details.txt");
        }
};

class Traveller {
        protected:
        string name;
        string address;
        string contact;
        string gender;

        public:
        Traveller(){
                name = "";
                address = "";
                gender = "";
                contact = "";
        }

        void setTDetails(){
                cout << "\n Name: ";
                getline(cin, name);
                cout << " Gender(M/F): ";
                getline(cin, gender);
                cout << " Address: ";
                getline(cin, address);
                cout << " Contact: ";
                getline(cin, contact);
        }

        void storeTDetails(int trip_id){
                ofstream fout;
                fout.open("traveller-details.txt", ios::app);
                fout << trip_id << " " << name << " " << gender << " " <<  address << " "
<< contact << endl;
                fout.close();
        }

        void deleteTravellerDetails(int delete_trip_id){
                ifstream fin;
                string readLine;
                ofstream fout;
```

```
		fin.open("traveller-details.txt", ios::in);
		if(fin.is_open())
		{
			while(getline(fin, readLine))
			{
				int fileTripId = 0;
				int size = readLine.length();
				char tripD[size+1];
				int temp = 0, i=0;
				strcpy(tripD, readLine.c_str());
				while((temp = tripD[i++]) != 32)
					fileTripId = fileTripId * 10 + (temp - 48);
				if(fileTripId != delete_trip_id)
				{
					fout.open("temp.txt", ios::app);
					fout<<readLine<<endl;
					fout.close();
				}
			}
			fin.close();
		}
		remove("traveller-details.txt");
		rename("temp.txt","traveller-details.txt");
	}
};

class Receipt {
	protected:
	int receipt_id;
	int sub_total;
	int tax;
	int grand_total;

	public:
	Receipt() {
		receipt_id = 0;
		sub_total = 0;
		tax = 0;
		grand_total = 0;
	}

	void generateReceipt(int trip_id, int transport_cost, int hotel_cost, int
noOfDays, string name){
		receipt_id = generateRandomId();

		sub_total = (hotel_cost * noOfDays) + transport_cost;
		tax = (sub_total * 18)/100;
		grand_total = sub_total + tax;
		cout << grand_total << endl;
		printReceipt(trip_id, sub_total, tax, grand_total, transport_cost,
hotel_cost, noOfDays, name);
	}

	void printReceipt(int trip_id, int sub_total, int tax, int grand_total, int
transport_cost, int hotel_cost, int noOfDays, string name){
		int hotelTotal = sub_total - transport_cost;

		clearScreen();
		string border(LINELENGTH, '^');
		cout << border;
		print(CENTRE, "Trip Booked", LINELENGTH);
		cout << endl << border;
```

```
            cout << "Receipt -> Loading...";
            wait(4000);
            clearScreen();
            print(LINELENGTH);
            cout << border;
            print(CENTRE, "Booking Receipt", LINELENGTH);
            cout << endl << border;
            cout << "Trip Id: " << trip_id << endl;
            cout << "Name: " << name << endl;
            cout << "Days(stay): " << noOfDays << endl;
            cout << endl << border;
            cout << left << setw(20) << "SR.NO" << left << setw(20) << "Description"
<< right << setw(40) << "Total Price";
            cout << endl << border;
            cout << left << setw(20) << "01" << left << setw(20) << "Transport" <<
right << setw(40) << setprecision(2) << transport_cost;
            cout << left << setw(20) << "02" << left << setw(20) << "Hotel" << right
<< setw(40) << setprecision(2) << hotelTotal;
            cout << left << setw(60) << " " << right << setfill('-') << setw(20) <<
"-";
            cout << left << setfill(' ') << setw(40) << " " << left << setw(20) <<
"SUBTOTAL:" << right << setw(20) << sub_total;
            cout << left << setw(40) << " " << left << setw(20) << "TAX(18%):" <<
right << setw(20) << tax;
            cout << left << setw(40) << " " << left << setw(20) << "GRAND TOTAL:" <<
right << setw(20) << grand_total;
            cout << endl << border;

        }

      void storeReceipt(int trip_id, int transport_cost, int hotel_cost, int
noOfDays, string name){

            generateReceipt(trip_id, transport_cost, hotel_cost, noOfDays, name);
            ofstream fout;
            fout.open("Receipt-details.txt", ios::app);
            fout << trip_id << " " << receipt_id  << " " << sub_total << " " <<
grand_total << endl;
            fout.close();

      }

      void deleteReceipt(int delete_trip_id){
            ifstream fin;
            string readLine;
            ofstream fout;

            fin.open("Receipt-details.txt", ios::in);
            if(fin.is_open())
            {
                  while(getline(fin, readLine))
                  {
                        int fileTripId = 0;
                        int size = readLine.length();
                        char tripD[size+1];
                        int temp = 0, i=0;
                        strcpy(tripD, readLine.c_str());
                        while((temp = tripD[i++]) != 32)
                              fileTripId = fileTripId * 10 + (temp - 48);
                        if(fileTripId != delete_trip_id)
                        {
                              fout.open("temp.txt", ios::app);
```

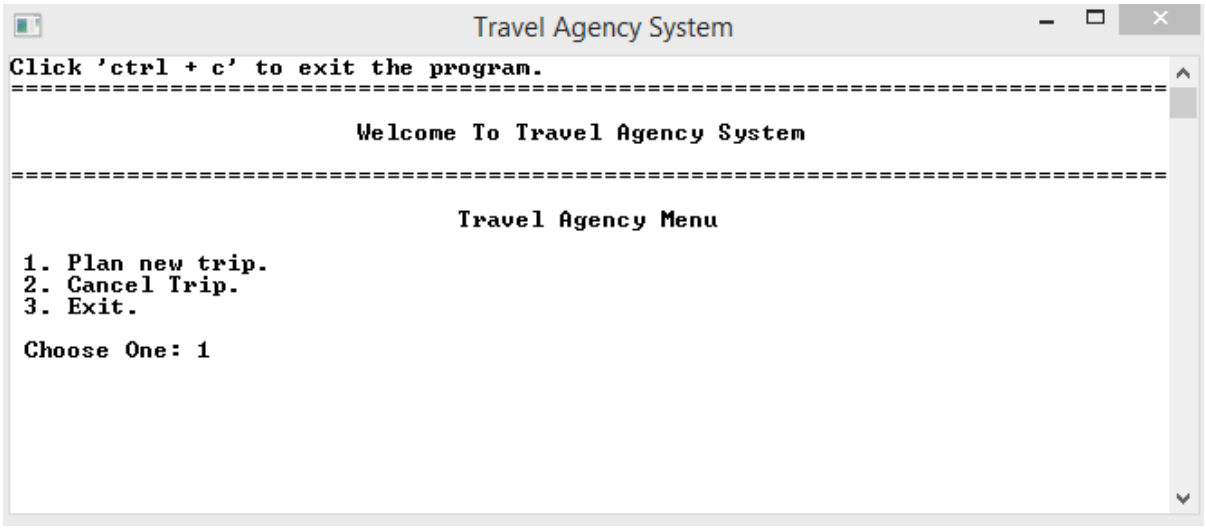
```
                                fout<<readLine<<endl;
                                fout.close();
                          }
                    }
                    fin.close();
             }
             remove("Receipt-details.txt");
             rename("temp.txt","Receipt-details.txt");
       }
};

class Trip:public Traveller, Transport, Hotel, Receipt{
       protected:
       int trip_id;
       string trip_start_date;
       string trip_end_date;
       string trip_to;
       string trip_from;
       int noOfDays;

       public:
       Trip() {
             trip_id = 0;
             trip_start_date = "";
             trip_end_date = "";
             trip_to = "";
             trip_from = "";
             noOfDays = 0;
       }

       void printTripMenu(int LINELENGTH){

             string border(LINELENGTH,'*');
             cout << endl << border;
             cout << "|" << left << setw(20) << " Trip option" << " | " << left <<
setw(55) << "Trip route" << right << "|";
             cout << border;
             cout << "|" << left << setw(20) << " 1" << " | " << left << setw(55) <<
"Goa to Bangalore" << right << "|";
             cout << "|" << left << setw(20) << " 2" << " | " << left << setw(55) <<
"Goa to Mumbai" << right << "|";
             cout << "|" << left << setw(20) << " 3" << " | " << left << setw(55) <<
"Goa to Delhi" << right << "|";
             cout << "|" << left << setw(20) << " 4" << " | " << left << setw(55) <<
"Goa to Shimla" << right << "|";
             cout << border << endl;

       }

       void setTripDestinations(int& tripOption){
             SelectTrip:  cout << " Enter trip option: ";
                                cin >> tripOption;

       cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
                                switch(tripOption){
                                      case 1: trip_from = "Goa";
                                                  trip_to = "Bengaluru";
                                      break;

                                      case 2: trip_from = "Goa";
                                                  trip_to = "Mumbai";
                                      break;
```

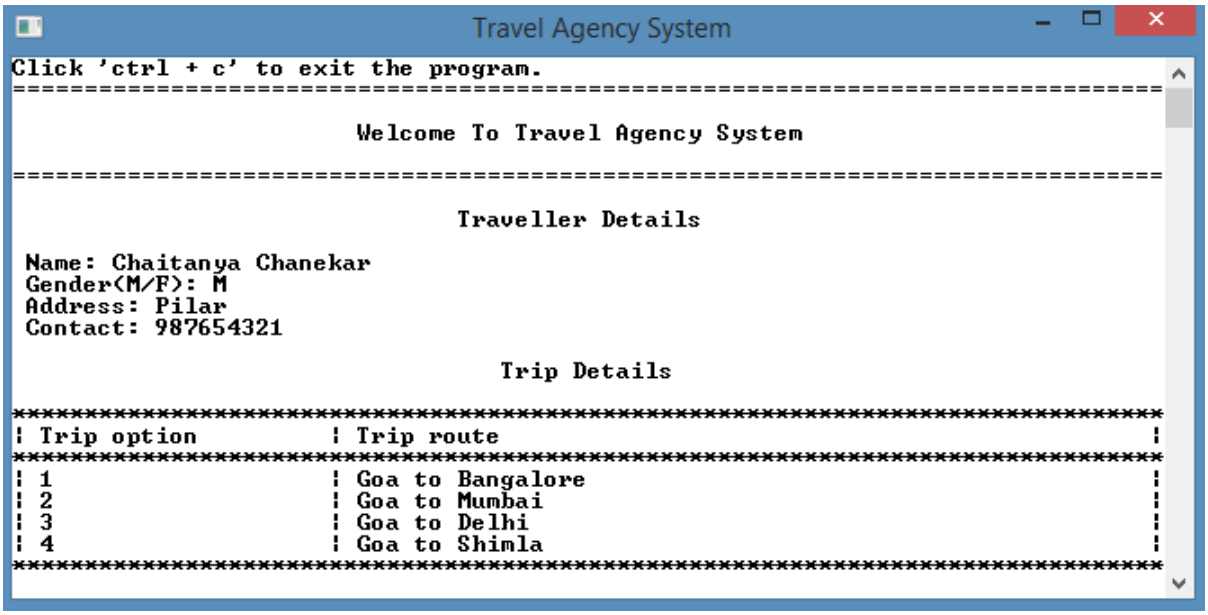
```
                                        case 3: trip_from = "Goa";
                                                trip_to = "Delhi";
                                        break;

                                        case 4: trip_from = "Goa";
                                                trip_to = "Shimla";
                                        break;
                                        default: cout << "\n Enter Correct option from
above..." << endl;
                                                 goto SelectTrip;
                                        break;
                                }
        }

        void getTripDetails(int LINELENGTH){
                int tripOption = 0;
                trip_id = generateRandomId();

                // set traveller details
                print(CENTRE,"Traveller Details", LINELENGTH);
                setTDetails();
                cout << endl;

                // set trip details
                print(CENTRE,"Trip Details", LINELENGTH);

                printTripMenu(LINELENGTH);
                setTripDestinations(tripOption);
                cout << " Trip start date: ";
                getline(cin, trip_start_date);
                cout << " Trip end date: ";
                getline(cin, trip_end_date);
                cout << " Days(stay): ";
                cin >> noOfDays;
                cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
                cout << endl;


                //set transport details
                print(CENTRE,"Transport Details", LINELENGTH);
                getTransportDetails(tripOption);
                cout << endl;

                // set hotel details
                print(CENTRE,"Hotel Details", LINELENGTH);
                getHotelDetails(LINELENGTH);


        }

        void storeTripDetails(){
                ofstream fout;
                fout.open("trip-details.txt", ios::app);
                fout << trip_id << " " << trip_start_date  << " " << trip_end_date << " "
<< trip_from << " " << trip_to << " " << noOfDays << endl;
                fout.close();
        }

        void bookTrip(int LINELENGTH){
                clearScreen();
                print(LINELENGTH); // prints the header
```

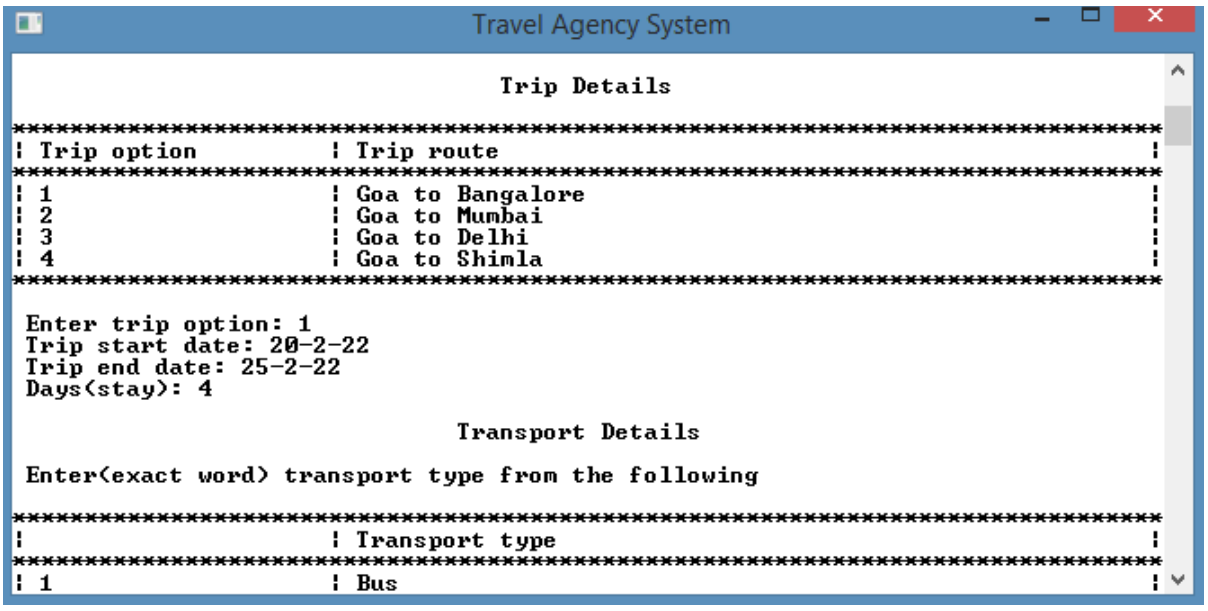
```
		getTripDetails(LINELENGTH); // get trip and other details

		storeTDetails(trip_id);
		bookTransport(trip_id);
		bookHotel(trip_id);
		storeTripDetails();
		storeReceipt(trip_id, transport_cost, hotel_cost, noOfDays, name);
	}

	void deleteTripDetails(int delete_trip_id){
		clearScreen();
		print(LINELENGTH);

		ifstream fin;
		string readLine;
		ofstream fout;

		fin.open("trip-details.txt", ios::in);
		if(fin.is_open())
		{
			while(getline(fin, readLine))
			{
				int fileTripId = 0;
				int size = readLine.length();
				char tripD[size+1];
				int temp = 0, i=0;
				strcpy(tripD, readLine.c_str());
				while((temp = tripD[i++]) != 32)
					fileTripId = fileTripId * 10 + (temp - 48);
				if(fileTripId != delete_trip_id)
				{
					fout.open("temp.txt", ios::app);
					fout<<readLine<<endl;
					fout.close();
				}
			}
			fin.close();
		}
		remove("trip-details.txt");
		rename("temp.txt","trip-details.txt");
	}

	void cancelTrip(){
		string border(LINELENGTH, '^');
		int delete_trip_id = 0;

		cout << "Enter trip id of the trip to be cancelled: ";
		cin >> delete_trip_id;
		cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');

		deleteTripDetails(delete_trip_id);
		cancelTransport(delete_trip_id);
		cancelHotel(delete_trip_id);
		deleteTravellerDetails(delete_trip_id);
		deleteReceipt(delete_trip_id);

		cout << border;
		print(CENTRE, "Trip Cancelled", LINELENGTH);
		cout << endl << border;
	}
};
```

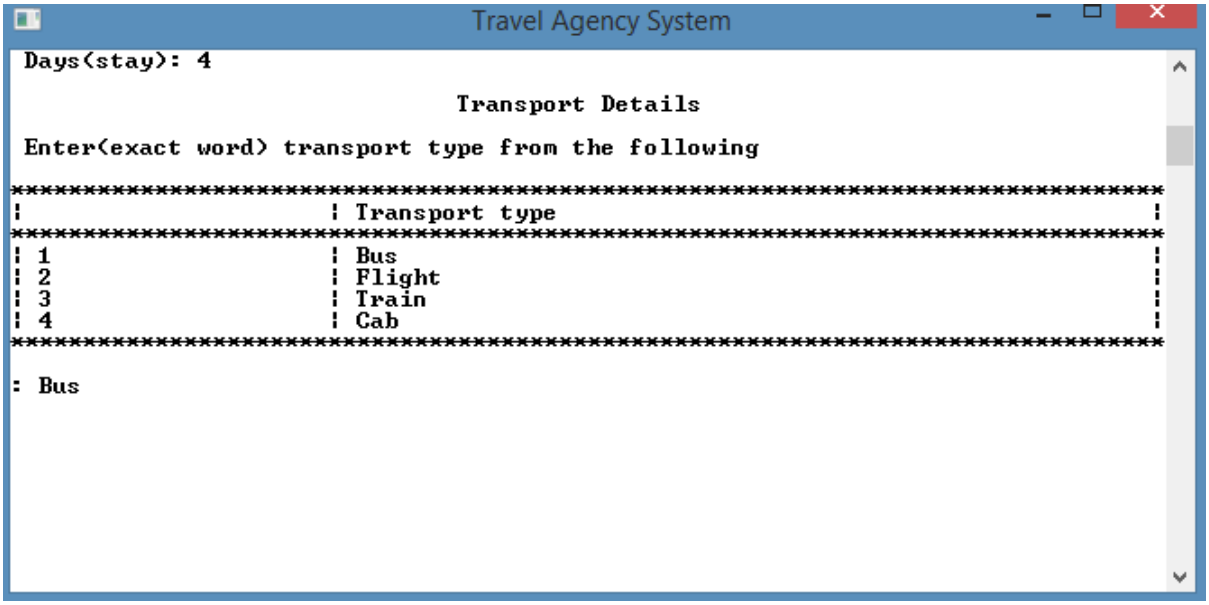
```
int main()
{
       clearScreen();
    system("title Travel Agency System");

    // Variables
    int chooseOneFromMenu = 0;
    char exitSurity;
    Trip tripObject;

       // menu
       MainMenu:
              clearScreen();
              print(LINELENGTH);
              printMainMenu(LINELENGTH);
           cin >> chooseOneFromMenu;
           cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');

    switch (chooseOneFromMenu)
    {
    case 1:
        tripObject.bookTrip(LINELENGTH);
        break;
    case 2:
        tripObject.cancelTrip(); // delete from the file
        break;
    case 3:
        ExitProgram:
               cout << "Program terminating. Are you sure? (Y/N): ";
               cin >> exitSurity;
               if (exitSurity == 'y' || exitSurity == 'Y') {
                   return 0;
               }else if (exitSurity == 'n' || exitSurity == 'N') {
                   clearScreen();
                   goto MainMenu;
               }else {
                   cout << "Next time choose after read the corresponding line." <<
endl;
                   goto ExitProgram;
               }
               break;
    default:
        cout << "Please choose between 1 - 3.";
        clearScreen();
        system("PAUSE");
        goto MainMenu;
        break;
    }
    system("PAUSE");
    goto MainMenu;
    return 0;
}
```

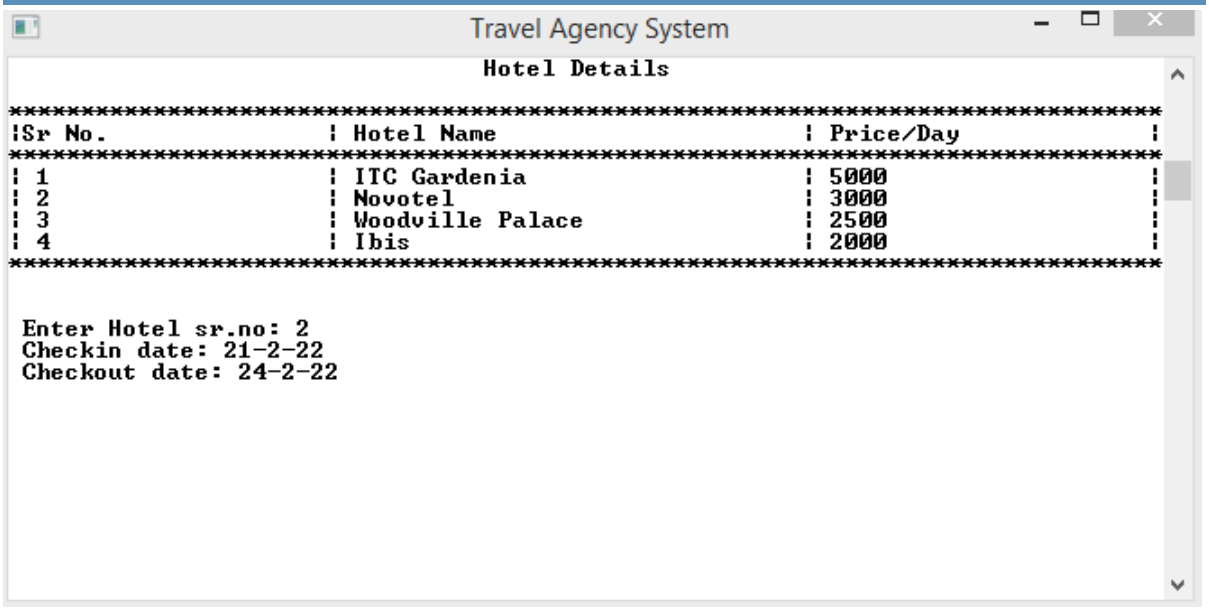
Output:

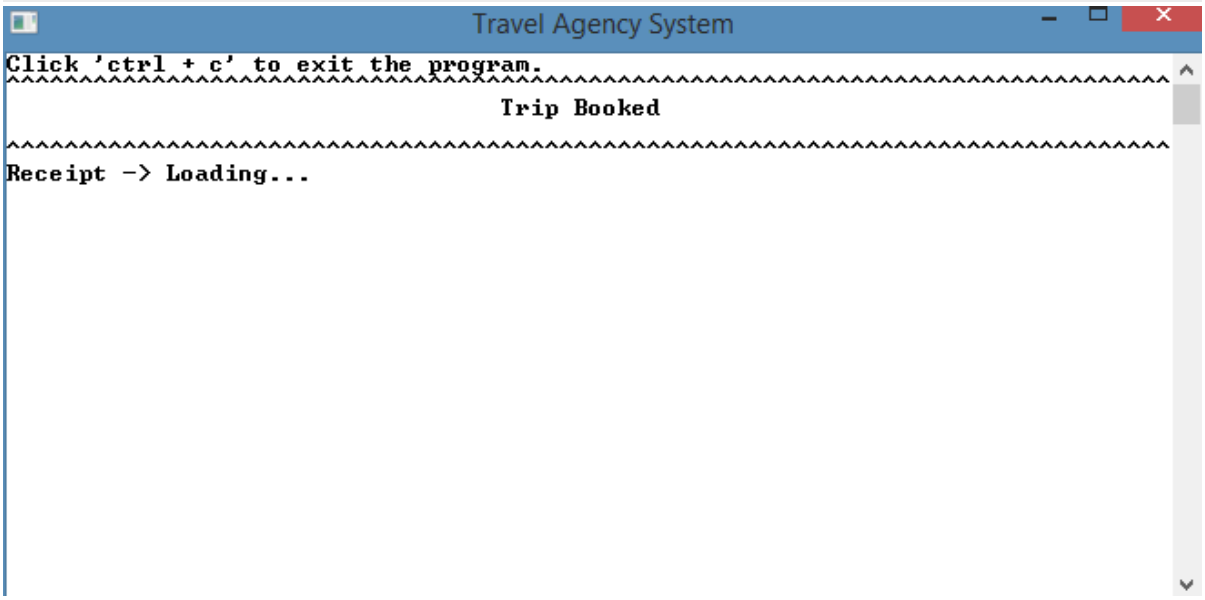

Travel Agency System
Click 'ctrl + c' to exit the program.
Welcome To Travel Agency System
Travel Agency Menu
1. Plan new trip.
2. Cancel Trip.
3. Exit.
Choose One: 1


Travel Agency System
Click 'ctrl + c' to exit the program.
Welcome To Travel Agency System
Traveller Details
Name: Chaitanya Chanekar
Gender(M/F): M
Address: Pilar
Contact: 987654321
Trip Details
Trip option	Trip route
1	Goa to Bangalore
2	Goa to Mumbai
3	Goa to Delhi
4	Goa to Shimla

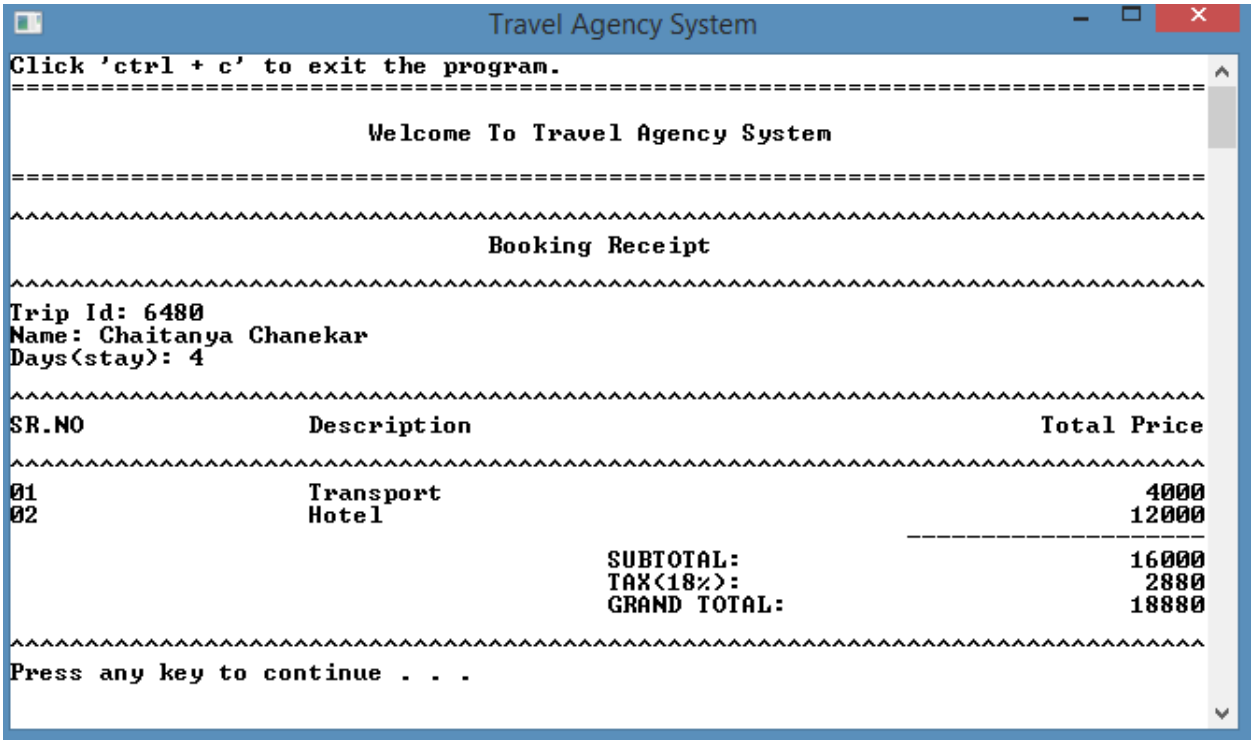

Travel Agency System
Trip Details
Trip option	Trip route
1	Goa to Bangalore
2	Goa to Mumbai
3	Goa to Delhi
4	Goa to Shimla
Enter trip option: 1
Trip start date: 20-2-22
Trip end date: 25-2-22
Days(stay): 4
Transport Details
Enter(exact word) transport type from the following
Transport type
1	Bus


Travel Agency System
Days(stay): 4
Transport Details
Enter(exact word) transport type from the following
	Transport type
1	Bus
2	Flight
3	Train
4	Cab
: Bus

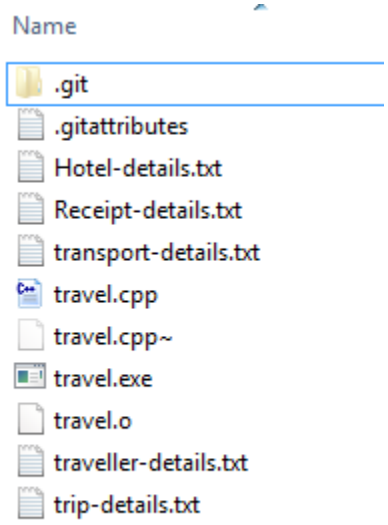

Travel Agency System
Hotel Details
Sr No.	Hotel Name	Price/Day
1	ITC Gardenia	5000
2	Novotel	3000
3	Woodville Palace	2500
4	Ibis	2000
Enter Hotel sr.no: 2
Checkin date: 21-2-22
Checkout date: 24-2-22

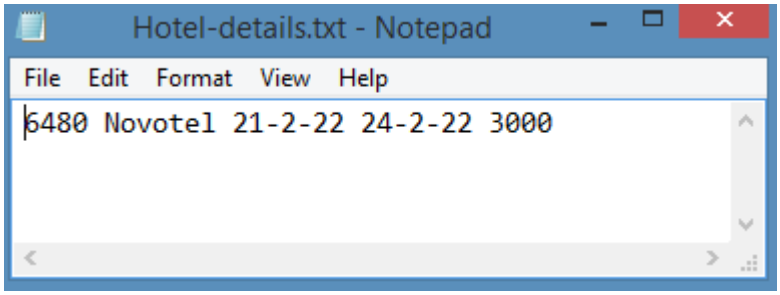

Travel Agency System
Click 'ctrl + c' to exit the program.
Trip Booked
Receipt -> Loading...

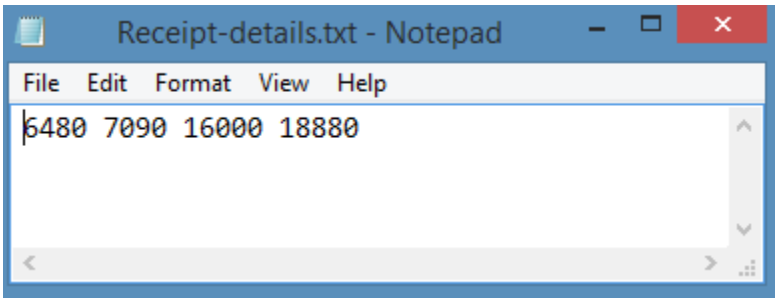
Receipt:

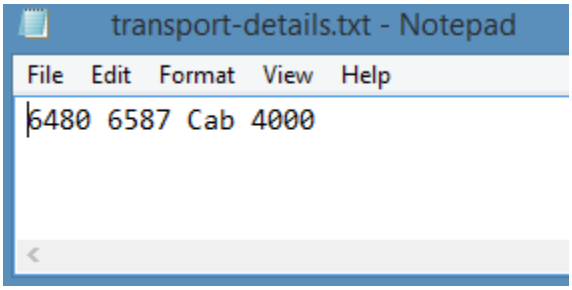


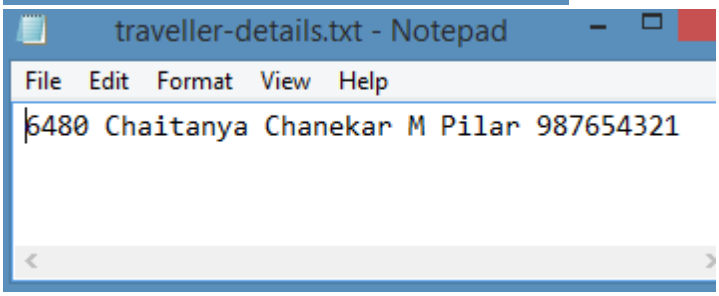
File system: after booking

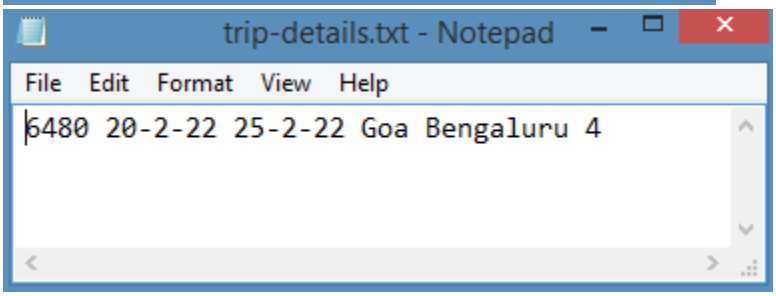




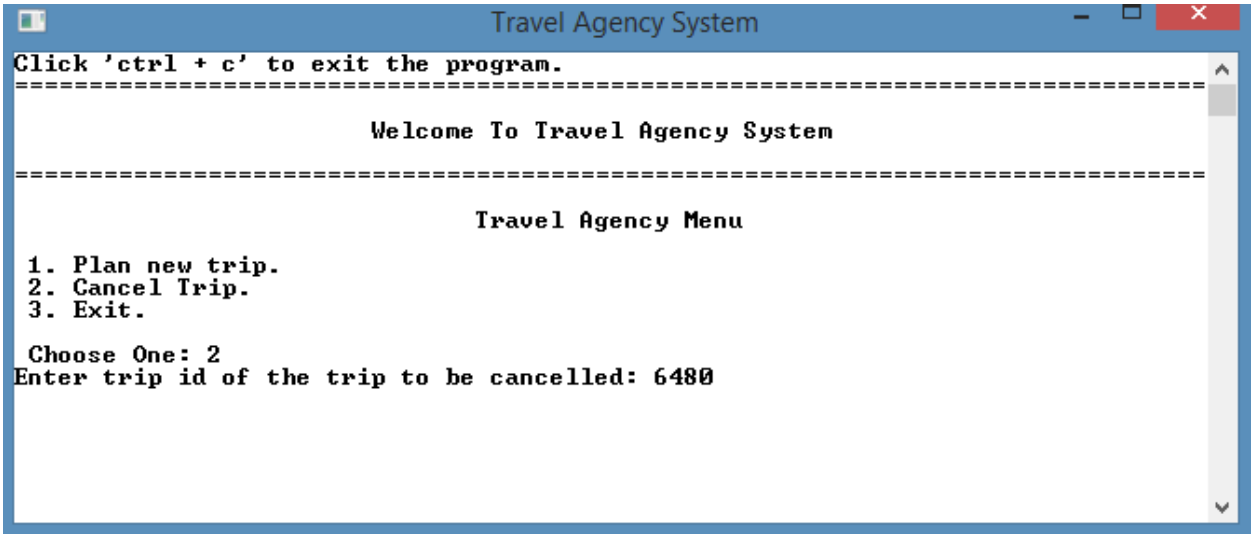


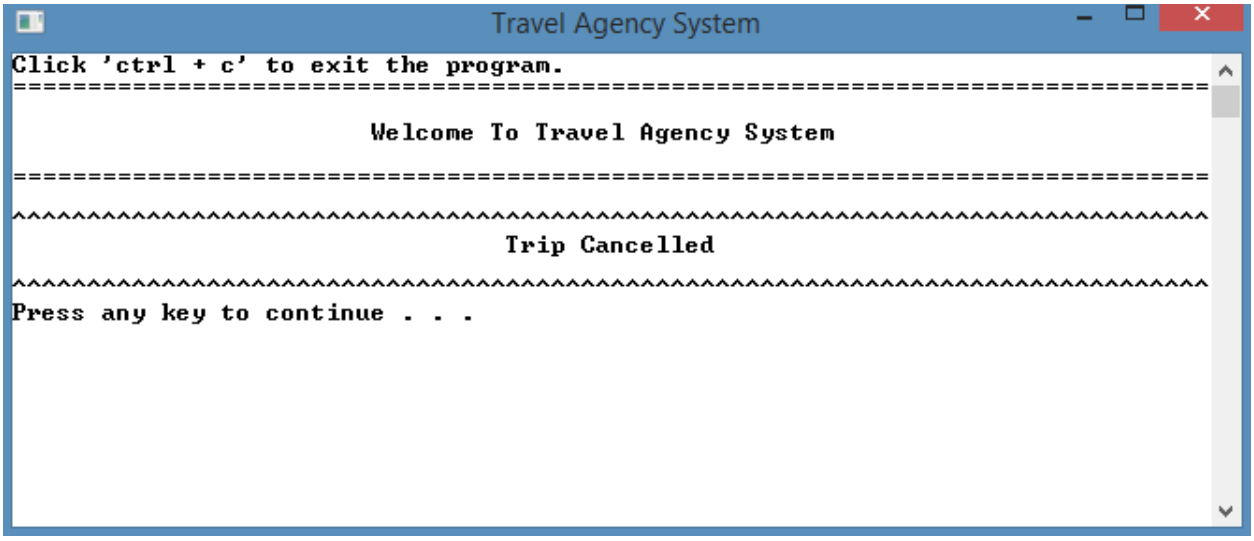






Canceling Trip:





File system after cancelling trip: