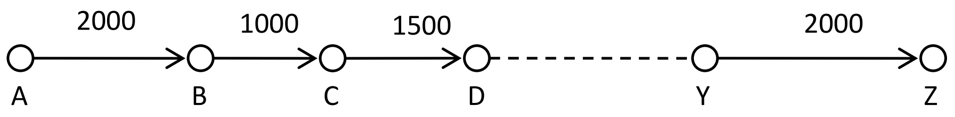
BLG252E - Object Oriented Programming
Homework-3
(This is the last homework)

Assignment Date : 08.12.2015
Due Date : 05.01.2016 at 18:00

The following drawing represents an air transportation line (one-direction) , in which letters are names of cities and numbers are distances (kilometer) between two cities. “A” is the first terminal city, “Z” is the last terminal city.



In this homework, you are given two text files.

“LINES.TXT” file contains the following data:
- line code
- line name
- line craft type
- number of cities it passes through
- sequence of (city code and city distance) pairs

“CITIES.TXT” file contains the following data:
- city code
- city name

Write C++ definitions for the **Line class** given in the UML diagram below**.**

```
Line
-----------------------------------------
+ line_code          :  int
+ line_name          :  string
+ line_craft_type    :  string
+ line_cities        :  vector<CityDistance>
+ average_speed      :  int
-----------------------------------------
+ Line (line_code, line_name, line_craft_type, line_cities)
+ print_schedule (start_city_code, end_city_code)  :  void
```

- line_cities is a **STL (Standard Template Library) vector.**
  This vector will contain the related CityDistance structs, whose C++ definition is given below:

```
struct CityDistance {
  int  city_code;
  int  city_distance;
};
```

- average_speed should be initialized by the Line class **constructor,** based on the following information.

Line craft type	Average speed (km/hour)
Boeing	600
Airbus	750
Jetfly	700
Lockheed	650

MAIN PROGRAM

Write a main C++ program to do followings:

- The main program should be written with a **try-catch block**.
- In global scope, declare a **STL map** of cities with <int, string> template parameters. The first template parameter (int) is the Key (city code). The second template parameter (string) is the Value (city name). Program should read **"CITIES.TXT"** file and initialize this map.
- Declare a **STL vector** of Line class objects. Program should read **"LINES.TXT"** file and initialize this vector.
- If any of the data files can not be opened, then the program must **throw** a related exception message.
- Display the user interface shown below, and perform user query operations.

First, the user interface should display a list of all avaliable lines, by using the **STL vector of lines**.

```
LINES MENU

Line_code  Line_name  Line_craft_type
1          LINE-1     Boeing
2          LINE-2     Airbus
3          LINE-3     Jetfly
4          LINE-4     Lockheed
5          LINE-5     Boeing
6          LINE-6     Jetfly

Enter Line Code : 4
```

Then, the user interface should display the names of all cities that are on the selected line, and ask user to select a start city code and end city code.

```
CITIES ON SELECTED LINE

City_code  City_name   Distance
13         Lancaster   0
11         Kimberley   2000
1          Apeldoorn   3000
26         Vinh        1000
10         Jaipur      2000
29         Yingkou     1000
17         Ogbomosho   3000
5          Datong      4000
14         Liaocheng   1000
20         Quilmes     3000

Enter Start City code and End City code : 26   5
```

- If user enters any invalid inputs, then the program must **throw** a related exception message.

PROGRAM OUTPUT

Main program should call the **print_schedule** function of the selected line object.
The **print_schedule** should generate and display a time schedule between the start city and the end city selected.

```
Departure_City   Departure_Time   Arrival_City   Arrival_Time   Distance(Km)   Travel Duration(Hours)
==============   ==============   ============   ============   ============   ======================
Vinh             18:00            Jaipur         21:00          2000           3
Jaipur           22:00            Yingkou        23:00          1000           1
Yingkou          24:00            Ogbomosho      04:00          3000           4
Ogbomosho        05:00            Datong         11:00          4000           6

TOTAL DISTANCE (Km) = 10000
TOTAL HOURS = 14
```

IMPORTANT:

- The first (earliest) departure time from a terminal city (first city in the line seqence) is 07:00.
- Arrival time at a city should be calculated based on the average speed of line and the distance between two cities.
- There should be 1 hour of waiting time at each arrived city, except the first terminal city and the last terminal city.
- For time calculations, you can round the hours so that the minutes will always be shown as zeros.
- If a time reaches 24, it should be set to zero after it.
- After performing a menu selection, there is no need to return to the main menu, program should stop.
- You may omit column alignments on user interface and output screens.