0. Introduction

In homework #2, more features are added to the Project Management Tool (PMT). These features include managing tasks, managing resources, adjusting task’s time, calculating project estimated cost, and some unit testing for your homework. First, the Tasks Management is a critical function for CRUD tasks, create the task dependency, and assign the resources. Second, the Resource Management is a function for CRUD resources. Third, Adjust task time is a mechanism that automatic scheduling all the related tasks. Fourth, we need to calculate the project cost to track the meeting budget. Finally, you have to do some unit testing for your homework.

In addition, a skeleton program is offered to implement homework #2. Therefore, you can choose either the skeleton program or your own program. In general, you are encouraged to implementing homework #2 based on your own program.

*CRUD: Create, read, update, and delete.

1. Object-based Programming

In the homework #1, we implement the Task and Resource Organizer in procedure-based programming. We will find the following problems for procedure-based programming:

1. Some of similar nature functions, the programmer has to keep track of all them.
2. If a variable of a structure is changed, we would maintain code hardly.

In order to solve the problems, we will change the program from procedure-based to object-based. Your program must satisfy Information Hiding principle, which means that:

1. Each data structure should be represented in class form.
2. Some member functions and data members should be private which are only accessed within a class.
3. Global variables and methods, expect the main method, should be avoided.
Figure 1 shows the class diagram for homework #2.

**Task** • **Item** • **People** classes encapsulate each information, and provides the get/set methods for data member.

- **TaskGroup** class knows **Task** and **TaskGroup** class, and provides the task management methods.

- **Resource** class knows **Item** and **People** class, and provides the resource management methods.

- **ProjectManager** class knows **TaskGroup** and **Resource** class, and provides base resource and tasks management methods, it plays a controller that dispatcher the work.
to the right objects.

FileHandler class provides project description file management methods.

You must need to change homework#1 functions to homework#2.

2. Tasks Management

Tasks Management should provide the functionality for adding, deleting, and modifying for tasks and task groups. There are new features needed to implement:

Add task: Add a new task into a taskGroup. The user is required to type task name, estimatedStart, estimatedEnd, and parent taskGroup id. The new task’s state must be 0. Because of the task id is unique, you must implement ProjectManager::createTask. The method will return a new task instance that has a unique id.

Delete task: The user is required to type task id. If the id exists, the task should be deleted(include task group’s sub task and dependency relation).

Modify task: Modify the task’s information. The user is required to type task id. If the id exists, the user will be required to type task name, estimatedStart, estimatedEnd, and parent taskGroup id.

Assign Resource: Assign the resource to a task’s resource reference list. The user is required to type the task id. If the id exists, the user is required to type the resource id that he wants to assign.

Remove Resource: Remove a task’s resource be assigned. The user is required to type the task id. If the id exists, the resource will be removed from the task’s resourceRef list.

Assign Dependency: Assign the task1 depending on the task2. The user is required to type task1’s and task2’s id.

Remove Dependency: Remove the dependency between with the task1 and the task2. If the task1 depends on the task2, first, the user is required to type the task1’s id. If the id exist, the user will enter the task2’s id.
The TaskGroup part are similar with Task part. The features include adding task, deleting task, modifying task, assigning/removing resource and dependency.

findTask and findTaskGroup methods may be implemented to support your other functional methods or unit testing.

3. Resource Management

Resource Management should provide the functionality for adding, deleting, and modifying for resource. Basically, there are new features needed to implement:

Add Item: Add a new Item to Resource. The user is required to type item name, startDate, endDate, price. Because of the item id is unique, you must implement ProjectManger::createItem. The method will return a new item instance that has a unique id.

Delete Item: The user is required to type a item id. If the id exists, the item will be removed from Resource. If the item has been assigned to task, you must remove the relation.

Modify Item: Modify the item’s information. The user is required to type a item id. If the id exists, the user will be required to type the item name, startDate, endDate, price.

The People part is similar with Item part. The features include adding people, deleting people, and modifying people.

findItem and findResource methods may be implemented to support your other functional methods or unit testing.

4. Adjust Task Time

When we like two task’s dependency relation, we need to adjust the task’s time. For example, like figure2, there are two tasks, task1 and task2. Task1’s end date is 20080930, and Task2’s start date is 20080927. If we let task2 depends on task1, the task2’s start date must be changed to 20081001 (skip the holiday). If the task2’s start date is after task1’s end date, we don’t need to adjust. The function may be used to the functions that would change the task’s time.
You need to implement `ProjectManager::adjustTaskTime`.

If a task group depends on another task, you must to check all of the task group’s task.

![Diagram](image)

**Figure 2**

5. Project Cost

If we want to know if we meet the budget goal, we must compute the project cost. The project cost is equal to sum of each task cost. **Before you compute the project cost, you must adjust the task’s time.**

For example:

- Task1’s duration is equal to 8 days, and the resource be assigned are R1 and R2.
- Task2’s duration is equal to 6 days, and the resource be assigned is R1.
- R1’s price is 30 per day, and R2’s price is 40 per day.

\[
\begin{align*}
\text{Task1 cost} &= 8 \times (30 + 40) = 560 \\
\text{Task2 cost} &= 6 \times 30 = 180 \\
\text{Project cost} &= 560 + 180 = 740
\end{align*}
\]

You need to implement `ProjectManager::computeProjectCost` method, and the `Resource::getPrice` method will support you.

6. Unit Testing

Begin with this homework, you have to do the unit testing for every following homework. We will provide a unit testing skeleton in this homework. The first test has 2 test cases: `ProjectManagerTestCase` and `FileHandlerTestCase`. Each test case has
a test suite that contains a number of test methods. There are 14 test methods in totally. These methods don’t cover all methods that be implemented. We encourage you to complete all methods that you implement.

<table>
<thead>
<tr>
<th>Test Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProjectManagerTestCase</td>
<td></td>
</tr>
<tr>
<td>testCreateTask</td>
<td>The new task’s id is unique</td>
</tr>
<tr>
<td>testCreateTaskGroup</td>
<td>The new taskGroup’s id is unique</td>
</tr>
<tr>
<td>testAddTask</td>
<td>Test adding task correctly</td>
</tr>
<tr>
<td>testAddTaskGroup</td>
<td>Test adding task group correctly</td>
</tr>
<tr>
<td>testDeleteTask</td>
<td>Test the task or group had been removed</td>
</tr>
<tr>
<td>testModifyTask</td>
<td>Test the information had been changed</td>
</tr>
<tr>
<td>testModifyTaskGroup</td>
<td>Test the information had been changed</td>
</tr>
<tr>
<td>testAddDependency</td>
<td>Test adding the dependency correctly</td>
</tr>
<tr>
<td>testRemoveDependency</td>
<td>Test the relation had been removed</td>
</tr>
<tr>
<td>testAssignResource</td>
<td>Test assigning the resource correctly</td>
</tr>
<tr>
<td>testRemoveAssignedResource</td>
<td>Test the relation has been removed</td>
</tr>
<tr>
<td>testAdjustTaskTime</td>
<td>Test the time is correct after adjusting</td>
</tr>
<tr>
<td>testComputeProjectCost</td>
<td>Test computing project cost correctly</td>
</tr>
<tr>
<td>FileHandlerTestCase</td>
<td></td>
</tr>
<tr>
<td>testLoadFile</td>
<td>Test loading TaskGroup and Resource correctly</td>
</tr>
</tbody>
</table>

Table 1: Test methods listing

Moreover, you should get used to the following items, so that make your unit tests more effective.

1. When a unit test fails, fix the code right away.
2. All unit tests pass but the code crashed: write a unit test that demonstrates the failure.
3. Maintain your unit tests! Keep them up to date with your architecture/specification changes.
4. Write your code so that it’s easy to write unit tests.
5. Refactor your code when necessary to make it more unit test friendly. This will almost always lead to better code as well.
6. Do not copy and paste from your application code that is needed to do setup.

7. Grading

After completing the homework#1, your program should provide the following
features:
1. Tasks Management (28%)
2. Resource Management (12%)
3. Adjust Task Time (15%)
4. Project Cost (7%)
5. Unit Testing (28%)
6. Document (5%)

Please write a C/C++ program to perform these features described above. The efficiency of your program is not a concern in this homework. The grading of this homework will be based only on the following criteria:
(a). Compile and run the program without errors.
(b). Read inputs in exactly the described format.
(c). Good coding style (the criteria will be listed on Table 1). (5%)
(d). Please don't modify the function interface (add parameter, delete parameter, or change the function name) in the skeleton. If you change the interface, you will get zero score.

<table>
<thead>
<tr>
<th>Abbrev.</th>
<th>Bad Coding Style</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Inadequate commenting</td>
</tr>
<tr>
<td>B</td>
<td>Names (identifiers) are not meaningful</td>
</tr>
<tr>
<td>C</td>
<td>Use of unnamed constant</td>
</tr>
<tr>
<td>D</td>
<td>Indentation does not indicate program structure</td>
</tr>
</tbody>
</table>

8. Homework submission

The homework needs to be submitted in both ways:

1. Electronic version
   All the related files in this homework have to compress in one ZIP file. The ZIP should include the following items:
   (1). Source code (the entire eclipse project).
   (2). Report (PDF or WORD).
   Please login the Open Cyber Classroom using your student ID from the following URL: http://mslin.ee.ntut.edu.tw/new/student/. You have to upload the electronic version to the web.

2. Printed version
   You have to write a brief summary for this homework that should include the following items:
   (1). The features that you finished in this homework.
   (2). Snapshots of program execution.
   (3). Source Code Listing as appendix.
(4). Measure the time that you spent in this homework. Please record the time precisely in the following table (Table 2).

Table 3 Measurement Example

<table>
<thead>
<tr>
<th>Data</th>
<th>Start</th>
<th>Stop</th>
<th>Interrupt</th>
<th>Hour</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>20080917</td>
<td>13:35</td>
<td>15:00</td>
<td></td>
<td>01:25</td>
<td>Run Skeleton program</td>
</tr>
<tr>
<td>20080917</td>
<td>16:00</td>
<td>17:00</td>
<td></td>
<td>01:00</td>
<td>Learning STL library</td>
</tr>
<tr>
<td>20080918</td>
<td>08:30</td>
<td>09:05</td>
<td></td>
<td>00:35</td>
<td>Writing sample for io</td>
</tr>
<tr>
<td>20080918</td>
<td>10:42</td>
<td>11:20</td>
<td></td>
<td>00:35</td>
<td>Writing parser</td>
</tr>
<tr>
<td>20080919</td>
<td>12:00</td>
<td>13:00</td>
<td></td>
<td>01:00</td>
<td>Writing parser</td>
</tr>
<tr>
<td>20080920</td>
<td>15:20</td>
<td>17:20</td>
<td></td>
<td>02:00</td>
<td>Writing ListAllTask()</td>
</tr>
<tr>
<td>20080920</td>
<td>09:00</td>
<td>09:44</td>
<td></td>
<td>00:44</td>
<td>Writing ListAllTaskDependency()</td>
</tr>
<tr>
<td>20080921</td>
<td>08:50</td>
<td>10:05</td>
<td></td>
<td>01:15</td>
<td>Writing ListAllTaskDuration()</td>
</tr>
<tr>
<td>20080921</td>
<td>13:20</td>
<td>15:46</td>
<td></td>
<td>02:26</td>
<td>Writing ListALLResource()</td>
</tr>
</tbody>
</table>