Program #1 (40 points)

Due Date

February 17, by 11:59pm

Submission

1. You must designate a submitter (one of the team members) to submit all required files to Canvas. The comment block on top of each Java file must include the names of all team members.
2. Your project with all Java files that allow the TAs or graders to run and grade, one copy per team. [25 points]
3. Time logs. **Everyone must submit a copy of this**. [5 points]
4. Test design document, one copy per team. [5 points]
5. Java docs, one copy per team. [5 points]

Program Description

In this program, you will implement a growable container class **Team**, and use it to hold a group of **TeamMembers**. The program will allow the user to create a project team without the restriction of how many team members you can have in a team. You should provide the following operations: **add**, **remove**, and **print** team members. Your program should read and process commands from the standard input, and display the results to the standard output. When the program runs, display `"Let's start a new team!"`, and your program MUST be able to process the following commands. All commands are case-sensitive, which means the commands with lowercase letters are invalid!

- **`A`** `command` – **add** a team member to the team. For example, `A Lily 2/3/2011,` which adds Lily as a team member who joined the company since 2/3/2011. You must check for duplication of the member. That is, if Lily is already in the team (given the same name and same date exist in the team), display "`Lily 2/3/2011 is already in the team.`". If Lily was successfully added, display "`Lily 2/3/2011 has joined the team.`". You MUST check if a date is valid.
- **`R`** command – **remove** a team member from the team. For example, `R Lily 2/3/2011`, which removes Lily from the team. If the remove was not successful, display "`Lily 2/3/2011 is not a team member.`", otherwise display "`Lily 2/3/2011 has left the team.`". You MUST check if a date is valid.
- **`P`** command – **output** all members in the team. If the team is empty, display "`We have 0 team members!`", otherwise display `"We have the following team members: "` followed by the list of team members, and `"-- end of the list --".`
- **`Q`** `command`—output all members in the team, display "`The team is ready to go!`", and terminate the program.

Program Requirement

1. You MUST follow the software development ground rules.
2. Given the sample input at the end of this document, your output MUST MATCH the output in the SAMPLE OUTPUT section.
3. You are required to log your times working on this project with the template provided. **You will lose up to 5 points** if the log is not submitted.
4. You MUST have the **testbed main** in the **TeamMembe**r class, and submit a Word document for the test cases. You will lose up to **5 points** if you failed to submit these files.
5. Each Java class must go in a separate file. **-2 points** if you put more than one Java class into a file.
6. Your program MUST handle bad commands! The commands are case-sensitive. You can use **switch case** statement and the **`charAt(0)`** method of the String class to check the commands. **-2 points** if you don't.

7. Download the Java files provided on Canvas (in a zip file.) You MUST finish the following classes, and you MUST finish all the methods in the Java files. **-2 points** for each method not finished or not used. You are NOT ALLOWED to use the Date class and any Java container classes from the Java library classes. **-10 points** if you do.

   (a) **Date class.** This class defines the properties of a date. You CANNOT add other data members to this class, you CANNOT change the signature of the methods, and you CANNOT do I/O in this class. **-2 points** if you do. You MUST check if a date is valid. For example, for the month of January, March, May, July, August, October and December, there are 31 days; for April, June, September and November, there are 30 days; February has 28 days in a normal year, and 29 days in a leap year. To determine whether a year is a leap year, follow these steps:

   Step 1. If the year is evenly divisible by 4, go to step 2. Otherwise, go to step 5.
   Step 2. If the year is evenly divisible by 100, go to step 3. Otherwise, go to step 4.
   Step 3. If the year is evenly divisible by 400, go to step 4. Otherwise, go to step 5.
   Step 4. The year is a leap year.
   Step 5. The year is not a leap year.

   You must write a **test bed main** for this class. This is a small main that you put at the bottom of your class that is used to test your class. You MUST include simple tests for ALL constructors and methods. **-1 points** for each method not tested.

   (b) **TeamMember class**. This class defines the properties and methods of a team member. You CANNOT add other data members to this class, you CANNOT change the signature of the methods, and you CANNOT do I/O in this class. **-2 points** if you did. You must write a **test bed main** for this class. You MUST include simple tests for ALL constructors and methods. **-1 points** for each method not tested. You MUST print out the results to ensure the correctness of the results. All test cases must be documented with a Word document. For example:

Test Case	Description	Sample Input	Expected result / output
1	Test the constructor and use toString() method to display the content of the object.	"Lily", "11/27/2011"	Lily 11/27/2011
2	Test the .equals() method	Case 1: "Lily", "11/27/2011" "Lily", "11/28/2011" Case 2: "Lily", "11/27/2011" "Lily", "11/27/2011"	Case 1: not equal Case 2: equal
...	…	…	…

You must follow the instructions in the "Test Specification" section of the Software Development Ground Rules.

   (c) **Team class**. This is the **container class** that defines a team and its operations. You CANNOT add other data members to this class, you CANNOT change the signature of the methods, and you CANNOT do I/O in this class, except the **print()** method. **-2 points** if you did. The add() method appends item to the end of the array, NO NEED to check duplicate here. The remove() method calls find() method and finds the index of the team member to be removed. Move the last item in the array to fill the slot and set the reference of the last item to null.

   (d) **ProjectManager class.** This class is the **user interface class** that handles **input** commands and **output** messages. Finish the **run() method** and all private methods. The main method in Prog2.java will call the run() method here. You MUST check if the team "contains" the member you're adding by calling the .**contains()** method. When adding or removing a member, you must check if a given date is valid.

Sample Input

```
R Lily 11/27/2011
r Lily 11/27/2011
p
P
A Lily 2/29/2011
A Lily 4/31/2011
A Lily 6/31/2011
A Lily 9/31/2011
A Lily 11/31/2011
A Lily 2/28/2011
A Chang 4/30/2008
A Stone 2/29/2000
A Brown 6/32/1995
A Brown 6/0/1995
A Brown 6/1/1995
P
A Mary 1/31/1997
A Mary 1/31/1997
A Eric 3/31/2011
R Eric 3/30/2011
P
R Eric 3/31/2000
R Eric 3/31/2011
P
A John 13/1/2016
A John 0/1/2016
A John 12/30/2012
A April 9/30/1999
A Chris 10/31/2013
A Eric 2/29/2012
R Mary 1/31/1997
P
A Chang 5/1/2012
A Chang 7/31/2012
a Chang 7/31/2012
R Chris 11/30/2012
P
R Lily 2/28/2011
R Chang 4/30/2008
R Eric 2/29/2012
R Stone 2/29/2000
q
Q
```

Sample Output

```
Lily 11/27/2011 is not a team member.
Command 'r' not supported!
Command 'p' not supported!
We have 0 team members!
2/29/2011 is not a valid date!
4/31/2011 is not a valid date!
6/31/2011 is not a valid date!
9/31/2011 is not a valid date!
11/31/2011 is not a valid date!
Lily 2/28/2011 has joined the team.
Chang 4/30/2008 has joined the team.
Stone 2/29/2000 has joined the team.
6/32/1995 is not a valid date!
6/0/1995 is not a valid date!
Brown 6/1/1995 has joined the team.
We have the following team members:
Lily 2/28/2011
Chang 4/30/2008
Stone 2/29/2000
Brown 6/1/1995
-- end of the list --
Mary 1/31/1997 has joined the team.
Mary 1/31/1997 is already in the team.
Eric 3/31/2011 has joined the team.
Eric 3/30/2011 is not a team member.
We have the following team members:
Lily 2/28/2011
Chang 4/30/2008
Stone 2/29/2000
Brown 6/1/1995
Mary 1/31/1997
Eric 3/31/2011
-- end of the list --
Eric 3/31/2000 is not a team member.
Eric 3/31/2011 has left the team.
We have the following team members:
Lily 2/28/2011
Chang 4/30/2008
Stone 2/29/2000
Brown 6/1/1995
Mary 1/31/1997
-- end of the list --
13/1/2016 is not a valid date!
0/1/2016 is not a valid date!
John 12/30/2012 has joined the team.
April 9/30/1999 has joined the team.
Chris 10/31/2013 has joined the team.
Eric 2/29/2012 has joined the team.
Mary 1/31/1997 has left the team.
We have the following team members:
Lily 2/28/2011
Chang 4/30/2008
Stone 2/29/2000
```

```
Brown 6/1/1995
Eric 2/29/2012
John 12/30/2012
April 9/30/1999
Chris 10/31/2013
-- end of the list --
Chang 5/1/2012 has joined the team.
Chang 7/31/2012 has joined the team.
Command 'a' not supported!
Chris 11/30/2012 is not a team member.
We have the following team members:
Lily 2/28/2011
Chang 4/30/2008
Stone 2/29/2000
Brown 6/1/1995
Eric 2/29/2012
John 12/30/2012
April 9/30/1999
Chris 10/31/2013
Chang 5/1/2012
Chang 7/31/2012
-- end of the list --
Lily 2/28/2011 has left the team.
Chang 4/30/2008 has left the team.
Eric 2/29/2012 has left the team.
Stone 2/29/2000 has left the team.
Command 'q' not supported!
We have the following team members:
Chang 7/31/2012
Chang 5/1/2012
April 9/30/1999
Brown 6/1/1995
Chris 10/31/2013
John 12/30/2012
-- end of the list --
The team is ready to go!
```