**PPT - Requirement 1**

The sales division of your company wants competent employees for presenting products to clients. The employees have submitted their presentation. Using your programming skills create an application to review and regulate the presentation using two domain classes namely PPT and Slide.  
  
**Requirement 1:**  
Let’s start off by creating two Slide objects and check whether they are equal.  
  
1.Create a **Slide**Class with the following attributes:

|  |  |
| --- | --- |
| **Member Field Name** | **Data Type** |
| \_number | int |
| \_name | string |
| \_content | string |
| \_layoutType | string |
| \_createdTime | DateTime |

2.Mark all attributes as private  
  
3.Generate appropriate properties  
  
4.Add a default constructor and a parameterized constructor to take in all attributes in the given order:   
**Slide( int \_number, string \_name, string \_content,string \_layoutType, DateTime \_createdTime ).**  
  
5.When the “**slide**” object is printed, it should display the following details: **[Override the ToString method]**  
**Print format:**  
Number:"\_number"  
Name:"\_name"  
Content:"\_content"  
Layout Type:"\_layoutType"  
Created Time:"\_createdTime"  
  
6.Two slides are considered same if they have the same name and number.Implement the logic in the appropriate function. (Case – Insensitive)**[Override the Equals method]**  
  
Create a class named as **Program**, which contains **Main** method. All the input operations are done in this method.  
It is also used to access the above class and its method to do this requirement.  
  
The input format consists of slide details separated by comma in the below order,  
**(\_number,\_name,\_content,\_layoutType,\_createdTime)**  
  
The Input to your program would be details of two slides, you need to display their details as given in "5th point(refer above)" and compare the two slides and display whether they are same or different.  
**Note:**There is an empty line between display statements. Print the empty lines in the **Main** method.  
  
**Problem Overview:**  
This requirement is compare the two slide objects(use overrided Equals method).  
First get the two slide details from user and convert that details to slide object.  
Finally compare that two object and if both are same, then print "**Slide 1 is same as Slide 2**", If both are different, then print "**Slide 1 and Slide 2 are different**". That comparision will occurs using above constraint.  
  
**Sample INPUT & OUTPUT 1:**

Enter slide 1 detail:

**4,Conclusion,Inferences obtained,Title and content,01-01-2018 15:00:00**

Enter slide 2 detail:

**4,Conclusion,Inferences obtained,Title and content,01-01-2018 15:00:00**

Slide 1:

Number: 4

Name: Conclusion

Content: Inferences obtained

Layout Type: Title and content

Created Time: 01-01-2018 15:00:00

Slide 2:

Number: 4

Name: Conclusion

Content: Inferences obtained

Layout Type: Title and content

Created Time: 01-01-2018 15:00:00

Slide 1 is same as Slide 2

**Sample INPUT & OUTPUT 2:**

Enter slide 1 detail:

**4,Conclusion,Inferences obtained,Title and content,01-01-2018 15:00:00**

Enter slide 2 detail:

**5,Conclusions,Inferences obtained,Title and content,01-01-2018 15:00:00**

Slide 1:

Number: 4

Name: Conclusion

Content: Inferences obtained

Layout Type: Title and content

Created Time: 01-01-2018 15:00:00

Slide 2:

Number: 5

Name: Conclusions

Content: Inferences obtained

Layout Type: Title and content

Created Time: 01-01-2018 15:00:00

Slide 1 and Slide 2 are different

**PPT - Requirement 2**

**Requirement 2:**  
Now we are gonna start creating a ppt and add slides to it. Start with creating a ppt and use menu-driven approach to add, remove, display details of the slide in the ppt.  
  
a)Create a class **Slide** with the following attributes: 

|  |  |
| --- | --- |
| **Member Field Name** | **Type** |
| \_number | int |
| \_name | string |
| \_content | string |
| \_layoutType | string |
| \_createdTime | DateTime |

Mark all the attributes as private.  
Include appropriate properties.

Add a default constructor and a parameterized constructor to take in all attributes in the given order:   
  
**Slide(int \_number, string \_name, string \_content, string \_layoutType, DateTime \_createdTime)**

b)Create a class **PPT** with the following attributes: 

|  |  |
| --- | --- |
| **Member Field Name** | **Type** |
| \_name | string |
| \_slideList | List<Slide> |

Mark all the attributes as private.  
Include appropriate properties.

Add a default constructor and a parameterized constructor to take in all attributes in the given order: **PPT(string \_name, List<Slide> \_slideList)**  
In constructor pass the **\_slideList** value as an empty list. Only one **PPT** will be present at a time.

c) Create the following static method in **Slide**class, 

|  |  |
| --- | --- |
| **Method Name** | **Description** |
| public static Slide CreateSlide(string detail) | This method accepts a string which contains slide details separated by commas. Split the details and create a slide object from the details and return it. |

The input format consists of slide details separated by comma in the below order,  
**\_number,\_name,\_content,\_layoutType,\_createdTime**

d) Create the following methods in **PPT**class, 

|  |  |
| --- | --- |
| **Method Name** | **Description** |
| public void AddSlideToPPT(Slide slide) | This method accepts a Slide object and adds the Slide to the Slide list of the current ppt. |
| public bool RemoveSlide(int number) | This method will get the slide's number and delete the slide with the specified number from the current ppt. If a slide with the given number found, delete the slide and return **true**. If a slide withthe number is not found return **false**. |
| public void DisplaySlides() | This method will display the slide list in the current ppt. If the slide list is empty display "**No slides to show"**, else display "**Slides in [ppt name]**" and display all the slide details in the specified format. Where [ppt name] specifies the name of the ppt. |

After deletion, if true is returned print "**Slide successfully deleted**", else print "**Slide not found in PPT**". After adding slide to the ppt, print "**Slide successfully added**".  
  
**Note:** The above print statements should be present in the main method.  
  
Create **Program** class with **Main**method.  
All the input operations are done in this method.  
It is also used to access the above class and its method to do this requirement.  
In main method get the name of the PPT and action to be performed from the user.  
Perform corresponding action to the choice of the user.  
  
**Problem Overview:**  
This requirement contains the menu driven,  
1.Add Slide  
2.Delete Slide  
3.Display Slides  
  
If the user select option-1, then get the slide details from the user(which is comma seperated), Split that string and convert to the slide object(Use CreateSlide method), and add that created slide to the PPT(use AddSlideToPPT method). After add that slide to the PPT then print "**Slide successfully added**".  
  
If the user select option-2, then get the slide number user want to delete, If slide found for that number, then remove that slide from ppt(Use RemoveSlide method) and print "**Slide successfully deleted**", If the slide is not found then print "**Slide not found in PPT**".

If the user select option-3, then display the ppt name and the slide list for that PPT( use the following format). If no slide in PPT then print "**No slides to show**". For this option use DisplaySlides method.

Override **ToString** method to print the sorted slides by following format.  
  
When the “**slide**” object is printed, it should display the following format  
**Print format:**  
**Console.WriteLine("{0,-10} {1,-15} {2,-30} {3,-20} {4,-10}", "Number","Name","Content","Layout type","Created time");**  
  
**Sample Input and Output:**  
Enter the name of the PPT:  
**Project1 PPT**  
1.Add Slide  
2.Delete Slide  
3.Display Slides  
4.Exit  
Enter your choice:  
**3**  
No slides to show  
1.Add Slide  
2.Delete Slide  
3.Display Slides  
4.Exit  
Enter your choice:  
**2**  
Enter the slide's number to be deleted:  
**1**  
Slide not found in PPT  
1.Add Slide  
2.Delete Slide  
3.Display Slides  
4.Exit  
Enter your choice:  
**1  
1,Title,Project title,Title only,12-12-2017 23:11:03**  
Slide successfully added  
1.Add Slide  
2.Delete Slide  
3.Display Slides  
4.Exit  
Enter your choice:  
**3**  
Slides in Project1 PPT  
Number     Name            Content                        Layout type          Created time  
1          Title           Project title                  Title only           12-12-2017 23:11:03  
1.Add Slide  
2.Delete Slide  
3.Display Slides  
4.Exit  
Enter your choice:  
**2**  
Enter the slide's number to be deleted:  
**1**  
Slide successfully deleted  
1.Add Slide  
2.Delete Slide  
3.Display Slides  
4.Exit  
Enter your choice:  
**2**  
Enter the slide's number to be deleted:  
**1**  
Slide not found in PPT  
1.Add Slide  
2.Delete Slide  
3.Display Slides  
4.Exit  
Enter your choice:  
**4**

**PPT - Requirement 3**

**Requirement 3:**  
   In this requirement, you need to review whether the content in the slide is spam or not.

a)Create a Class **Program**with the following static methods:

|  |  |
| --- | --- |
| **Method** | **Description** |
| static bool IsSpam(string content) | Review the content based on the rules given below. Returns **true**if content is valid else return **false** |

b)While reviewing content follow the below rules,  
  
1. The same character should not appear more than twice consecutively in the content.  
Eg (aaa is spam)

2.The content should contain more than 15 characters excluding spaces.

**Note:**Print "**Content is a spam**" if content is spam else print "**Content is not a spam**".  
            All the above print statements are present in the main method.

Create a class **Program** with **Main** method to implement the functionality.  
All the input and output operations are done in this method.  
  
**Sample Input and Output 1:**  
Enter the content to be reviewed:  
**This chapter describes the hardware and software requirements of Oracle Communications Data Model**  
Content is not a spam  
  
**Sample Input and Output 2:**  
Enter the content to be reviewed:  
**Progress Ganttt chart**  
Content is a spam

**PPT - Requirement 4**

**Requirement 4:**  
In this requirement develop a feature in which you can search a List of Slides by layoutType,createdTime.  
  
a) Create a class **Slide** with the following attributes:

|  |  |
| --- | --- |
| **Member Field Name** | **Type** |
| \_number | int |
| \_name | string |
| \_content | string |
| \_layoutType | string |
| \_createdTime | DateTime |

Mark all the attributes as private.  
Include appropriate properties.  
Add a default constructor and a parameterized constructor to take in all attributes in the given order:   
**Slide(int \_number, string \_name, string \_content, string \_layoutType, DateTime \_createdTime)**  
  
b) Create a class **SlideBO**with the following methods,

|  |  |
| --- | --- |
| **Method Name** | **Description** |
| public List<Slide> FindSlide(List<Slide> slideList,string type) | This method accepts a list of slides and type as arguments and returns a list of slides that match with given type. |
| public List<Slide> FindSlide(List<Slide> slideList,DateTime createdTime) | This method accepts a list of slides and created time as arguments and returns a list of slides that matches with the given time. |

The slide details should be given as a comma-separated value in the below order,  
**\_number,\_name,\_content,\_layoutType,\_createdTime**  
  
Create **Program** class with **Main** method. In main method get the  number of slides and respective slide details from user.  
Get the type of search and perform corresponding action.  
  
**Problem Overview:**  
The first line of input contains an int, that corresponds to the number of slides n.  
The next n line of input consist of a string, that corresponds to the slide details(which is comma seperated). Split that string and create a slide object and add to the slide list (which is maintained in **Main** method).  
  
This requirement contains menu driven,  
1.By Layout Type  
2.By Created Time  
  
If the user select option-1, then get the layout type from user and filter the slide list by \_layoutType( Use FindSlide(List, string)), and display that slide list by following format.  
  
If the user select option-2, then get the created time from user and filter the slide list by \_createdTime( Use FindSlide(List, DateTime)), and display that slide list by following format.  
  
Override **ToString** method to print the sorted slides by following format.  
  
When the “slide” object is printed, it should display the following details

Print format:  
**Console.WriteLine("{0,-10} {1,-15} {2,-30} {3,-20} {4,-10}", "Number","Name","Content","Layout Type","Created Time");**  
  
  
**Note:**The slide lists are displayed in the main method.  
              If any other choice is selected, display "**Invalid choice**"  
              If search detail is not found, display "**No such slide is present**"

**Sample Input and Output 1:**  
  
Enter the number of slides:  
**3**  
**2,Names,Participants' name,Section header,22-02-2018 18:20:00**  
**4,Conclusion,Inferences obtained,Title and content,01-01-2018 15:00:00**  
**7,Requirement,Hardware and software req,Comparison,20-03-2018 18:00:30**  
Enter a search type:  
1.By Layout Type  
2.By Created Time  
**1**  
Enter the Layout Type:  
**Comparison**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number | Name | Content | Layout Type | Created Time |
| 7 | Requirement | Hardware and software req | Comparison | 20-03-2018 18:00:30 |

**Sample Input and Output 2:**  
  
Enter the number of slides:  
**3**  
**2,Names,Participants' name,Section header,22-02-2018 18:20:00**  
**4,Conclusion,Inferences obtained,Title and content,01-01-2018 15:00:00**  
**7,Requirement,Hardware and software req,Comparison,20-03-2018 18:00:30**  
Enter a search type:  
1.By Layout Type  
2.By Created Time  
**2**  
Enter the Created Time:  
**01-01-2018 15:00:00**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number | Name | Content | Layout Type | Created Time |
| 4 | Conclusion | Inferences obtained | Title and content | 01-01-2018 15:00:00 |

**Sample Input and Output 3:**  
  
Enter the number of slides:  
**3**  
**2,Names,Participants' name,Section header,22-02-2018 18:20:00**  
**4,Conclusion,Inferences obtained,Title and content,01-01-2018 15:00:00**  
**7,Requirement,Hardware and software req,Comparison,20-03-2018 18:00:30**  
Enter a search type:  
1.By Layout Type  
2.By Created Time  
**20**  
Invalid choice

**PPT - Requirement 5**

**Requirement 5:**  
  
In this requirement, you need to sort the list of slides based on name, createdTime or number.   
  
a) Create a Class **Slide** with the following attributes:

|  |  |
| --- | --- |
| **Member Field Name** | **Type** |
| \_number | int |
| \_name | string |
| \_content | string |
| \_layoutType | string |
| \_createdTime | DateTime |

Mark all the attributes as private.  
Include appropriate Properties.  
 Add a default constructor and a parameterized constructor to take in all attributes in the given order:   
**Slide(int \_number, string \_name, string \_content,string \_layoutType, DateTime \_createdTime)**

b) Create the following static methods in the **Slide** class,

|  |  |
| --- | --- |
| **Method Name** | **Description** |
| static Slide CreateSlide(string detail) | This method accepts a String. The slide detail separated by commas is passed as the argument. Split the details and create a slide object and return it. |

The slide details should be given as a comma-separated value in the below order,  
**(\_number,\_name,\_content,\_layoutType,\_createdTime)**  
  
c) The **Slide** class should implement the I**Comparable** interface which sorts the Slide list based on name attributes.While comparing, all the name attributes in the list are unique [Case Sensitive. Uppercase letters followed Lowercase].  
  
d) Create a class **CreatedTimeComparer** which implements **IComparer** interface and sorts the Slide list based on createdTime. While comparing, all the createdTime attributes in the list are unique.

e) Create a class **NumberComparer** which implements **IComparer**interface and sorts the Slide list based on the number. While comparing, all the number attributes in the list are unique.  
  
Get the number of Slides and Slide details and create a Slide list. Sort the Slide according to the given option and display the list.  
  
Create a **Program** class with the **Main** method to implement the functionality.  
This method is used to access above classes and its method to do this requirement.  
All the input and output operations are done in this method.  
  
**Problem Overview:**  
The first line of input contains an int, that corresponds to the number of slides n.  
The next n line of input consist of a string, that corresponds to the slide details(which is comma seperated). Split that string and create a slide object( Use CreateSlide method ) and add to the slide list (which is maintained in **Main** method).

This requirement contains menu driven,

1.Sort by name

2.Sort by number

3.Sort by created time

If the user select option-1, then sort that slide list using slide \_name(by ascending order).  
If the user select option-2, then sort that slide list using slide \_number(by ascending order) use **NumberComparer** class to sort.  
If the user select option-3, then sort that slide list using slide \_createdTime(by ascending order) use **CreatedTimeComparer** class to sort.

Override **ToString** method to print the sorted slides by following format.  
  
When the “slide” object is printed, it should display the following details.  
**Print format:**  
**Console.WriteLine(String.Format("{0,-10}{1,-15}{2,-30}{3,-20}{4,-10}", "Number", "Name", "Content", "Layout type", "Created time"));**  
  
**Sample Input and Output 1:**  
Enter the number of the slides:

**4**

**2,Names,Participants' name,Section header,22-02-2018 18:20:00**

**1,Title,Project title,Title only,12-12-2017 23:11:03**

**7,Requirement,Hardware and software req,Comparison,20-03-2018 18:00:30**

**4,Conclusion,Inferences obtained,Title and content,01-01-2018 15:00:00**

Enter a type to sort:

1.Sort by name

2.Sort by number

3.Sort by created time

**1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number | Name | Content | Layout type | Created time |
| 4 | Conclusion | Inferences obtained | Title and content | 01-01-2018 15:00:00 |
| 2 | Names | Participants' name | Section header | 22-02-2018 18:20:00 |
| 7 | Requirement | Hardware and software req | Comparison | 20-03-2018 18:00:30 |
| 1 | Title | Project title | Title only | 12-12-2017 23:11:03 |

**Sample Input and Output 2:**

Enter the number of the slides:

**4**

**2,Names,Participants' name,Section header,22-02-2018 18:20:00**

**1,Title,Project title,Title only,12-12-2017 23:11:03**

**7,Requirement,Hardware and software req,Comparison,20-03-2018 18:00:30**

**4,Conclusion,Inferences obtained,Title and content,01-01-2018 15:00:00**

Enter a type to sort:

1.Sort by name

2.Sort by number

3.Sort by created time

**2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number | Name | Content | Layout type | Created time |
| 1 | Title | Project title | Title only | 12-12-2017 23:11:03 |
| 2 | Names | Participants' name | Section header | 22-02-2018 18:20:00 |
| 4 | Conclusion | Inferences obtained | Title and content | 01-01-2018 15:00:00 |
| 7 | Requirement | Hardware and software req | Comparison | 20-03-2018 18:00:30 |

**Sample Input and Output 3:**

Enter the number of the slides:

**4**

**2,Names,Participants' name,Section header,22-02-2018 18:20:00**

**1,Title,Project title,Title only,12-12-2017 23:11:03**

**7,Requirement,Hardware and software req,Comparison,20-03-2018 18:00:30**

**4,Conclusion,Inferences obtained,Title and content,01-01-2018 15:00:00**

Enter a type to sort:

1.Sort by name

2.Sort by number

3.Sort by created time

**3**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number | Name | Content | Layout type | Created time |
| 1 | Title | Project title | Title only | 12-12-2017 23:11:03 |
| 4 | Conclusion | Inferences obtained | Title and content | 01-01-2018 15:00:00 |
| 2 | Names | Participants' name | Section header | 22-02-2018 18:20:00 |
| 7 | Requirement | Hardware and software req | Comparison | 20-03-2018 18:00:30 |
| **PPT - Requirement 6**    **Requirement 6:**  In this requirement, given a list of Slides, you need to find the number of slides on a layoutType using SortedDictionary.  a) Create a Class **Slide**with the following attributes:   |  |  | | --- | --- | | **Member Field Name** | **Type** | | \_number | int | | \_name | string | | \_content | string | | \_layoutType | string | | \_createdTime | DateTime |   Mark all the attributes as private. Include appropriate properties. Add a default constructor and a parameterized constructor to take in all attributes in the given order:  **Slide(int \_number, string \_name, string \_content, string \_layoutType, DateTime \_createdTime)**  b) Create the following static methods in the Slide class,   |  |  | | --- | --- | | **Method Name** | **Description** | | static SortedDictionary<string,int> CalculateLayoutCount(List<Slide> list) | This method accepts a list of Slide as arguments and returns a SortedDictionary with the LayoutType as key and number of slides on the layoutType as value and returns the SortedDictionary. |   In the SortedDictionary have the layoutType as key and Count the number of slides on the layoutType and keep the number of Slides as value. Print the value sorted by LayoutType.  Create **Program** class with **Main** method. All the input and output operations are done in this method. The Slide details should be given as a comma-separated value in the below order, \_number,\_name,\_content,\_layoutType,\_createdTime  **Problem Overview:** The first line of input contains an int, that corresponds to the number of slides n. The next n line of input consist of a string, that corresponds to the slide details(which is comma seperated). Split that string and create a slide object and add to the slide list (which is maintained in **Main** method).  Calculate the layout type count(number of occuring the each layout type) use CalculateLayoutCount method. Please refer the sample input and output and also the following format to print format.  Print format: **Console.WriteLine("{0,-20} {1,-10}","Layout type","Count");**  **Sample Input and Output 1:**  Enter the number of slides:  **5**  **8,Survey,Literature survey for papers,Blank,05-05-2017 12:00:00**  **14,References,website links,Blank,07-02-2018 11:50:30**  **4,Conclusion,Inferences obtained,Title and content,01-01-2018 15:00:00**  **7,Requirement,Hardware and software req,Comparison,20-03-2018 18:00:30**  **11,Progress,Progress Gnantt chart,Title and content,06-01-2018 12:12:00**   |  |  | | --- | --- | | Layout type | Count | | Blank | 2 | | Comparison | 1 | | Title and content | 2 | |  |  |  |  |