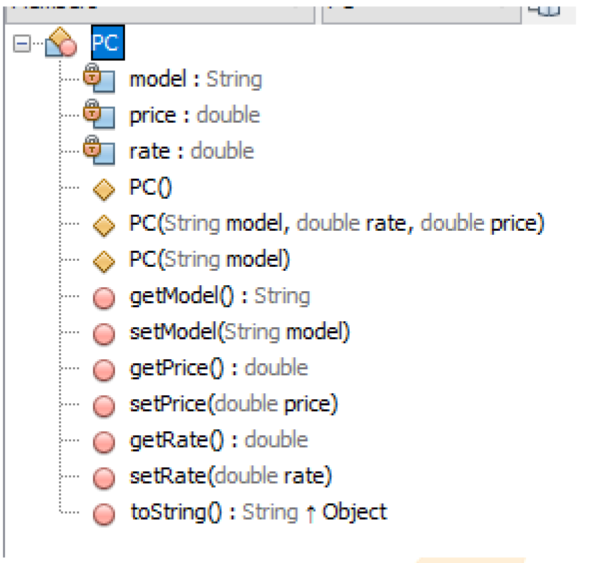
LAB 5 PRO192

Create a class name PC as in the image below.

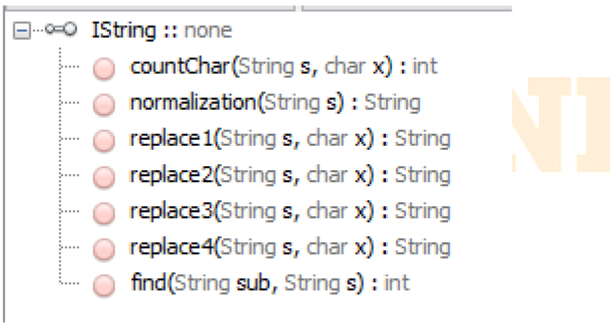


```
PC
  model : String
  price : double
  rate : double
  PC()
  PC(String model, double rate, double price)
  PC(String model)
  getModel() : String
  setModel(String model)
  getPrice() : double
  setPrice(double price)
  getRate() : double
  setRate(double rate)
  toString() : String ↑ Object
```

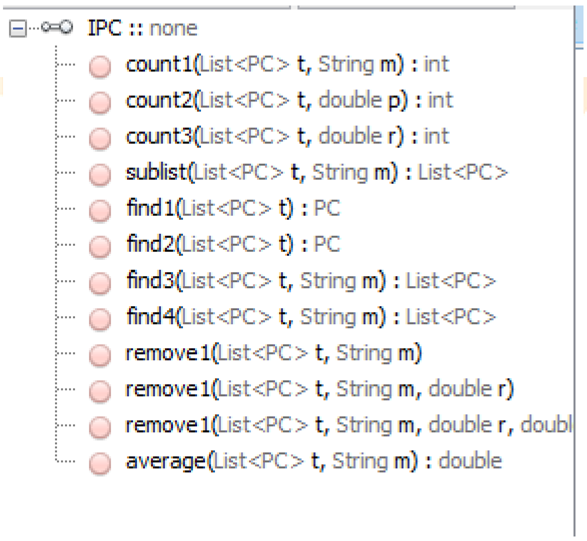
The getModel() method returns a string by converting the first two characters of the model to uppercase and the others to lowercase.

toString() return a string in format: (model,rate,price)

There are two interfaces that have been compiled to bytecode, you can use it without creating a .java file

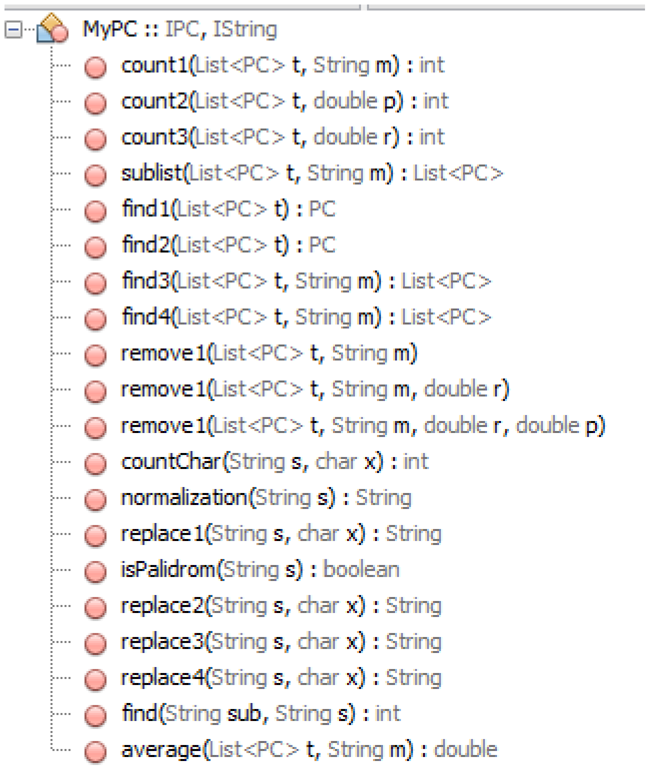


```
IString :: none
  countChar(String s, char x) : int
  normalization(String s) : String
  replace1(String s, char x) : String
  replace2(String s, char x) : String
  replace3(String s, char x) : String
  replace4(String s, char x) : String
  find(String sub, String s) : int
```



```
IPC :: none
  count1(List<PC> t, String m) : int
  count2(List<PC> t, double p) : int
  count3(List<PC> t, double r) : int
  sublist(List<PC> t, String m) : List<PC>
  find1(List<PC> t) : PC
  find2(List<PC> t) : PC
  find3(List<PC> t, String m) : List<PC>
  find4(List<PC> t, String m) : List<PC>
  remove1(List<PC> t, String m)
  remove1(List<PC> t, String m, double r)
  remove1(List<PC> t, String m, double r, doubl
  average(List<PC> t, String m) : double
```

Create a class name MyPC that implements the above two interfaces then complete all abstract methods (all methods are described as below).



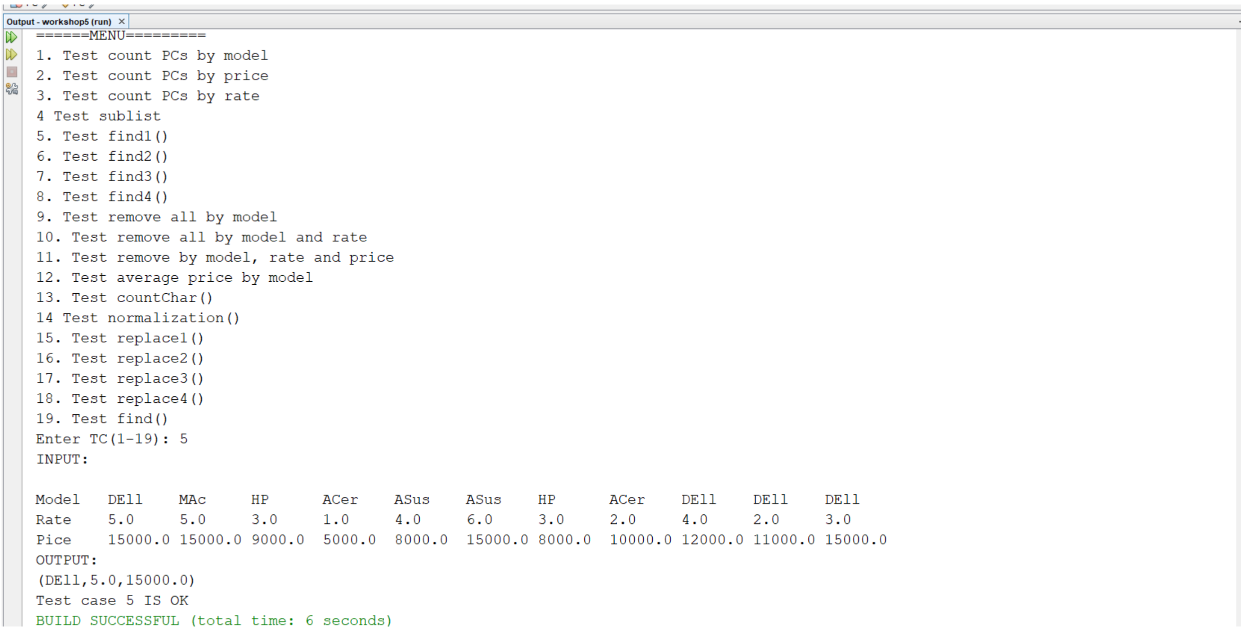
```
MyPC :: IPC, IString
    count1(List<PC> t, String m) : int
    count2(List<PC> t, double p) : int
    count3(List<PC> t, double r) : int
    sublist(List<PC> t, String m) : List<PC>
    find1(List<PC> t) : PC
    find2(List<PC> t) : PC
    find3(List<PC> t, String m) : List<PC>
    find4(List<PC> t, String m) : List<PC>
    remove1(List<PC> t, String m)
    remove1(List<PC> t, String m, double r)
    remove1(List<PC> t, String m, double r, double p)
    countChar(String s, char x) : int
    normalization(String s) : String
    replace1(String s, char x) : String
    isPalidrom(String s) : boolean
    replace2(String s, char x) : String
    replace3(String s, char x) : String
    replace4(String s, char x) : String
    find(String sub, String s) : int
    average(List<PC> t, String m) : double
```

```
public class MyPC implements IPC, IString {

   @Override
   public int F1_count1(List<PC> t, String m) {
      //count how many PCs in the list t which have a model equal to m (ignoring
case sensitivity)

      }

   @Override
   public int F2_count2(List<PC> t, double p) {
      //  count how many PCs in the list t which have a price = p
        }

   @Override
   public int F3_count3(List<PC> t, double r) {
      // count how many PCs in the list t which have a rate= r

   }

   @Override
   public List<PC> F4_sublist(List<PC> t, String m) {
//returns a sublist of List t where PC has a model equal to m (ignoring case
sensitivity)
```

```
    }
    @Override
    public PC F5_find1(List<PC> t) {
    //find the first PC in the list t has maximum price and model equal DELL.
    }
    @Override
    public PC F6_find2(List<PC> t) {
   //       find the last PC in the list t has maximum price and model equal DELL
(ignoring case sensitivity).

    }
    @Override
    public List<PC> F7_find3(List<PC> t,String m) {
   // find the all PC in the list t has minimum rate and model equal m(ignoring case
sensitivity).
    }
    @Override
    public List<PC> F8_find4(List<PC> t,String m) {
     // find the all PC in the list t has maximum rate and model equal m(ignoring case
sensitivity).
    }
    @Override
    public void F9_remove1(List<PC> t, String m) {
       //       remove all PCs with model equal to m (ignoring case sensitivity)

    }
    @Override
    public void F10_remove1(List<PC> t, String m, double r) {
       //       remove all the PC which have model equal m, rate = r
    }

    @Override
    public void F11_remove1(List<PC> t, String m, double r, double p) {
      //       remove all the PC which have model equal m, rate = r and price = p.

    }
   @Override
    public double F12_average(List<PC> t, String m) {
//return average price of all PC have model as m  (ignoring case sensitivity)
}
```

```
    @Override
    public int F13_countChar(String s, char x) {
        //return the number of character x in String s
}
    @Override
    public String F14_normalization(String s) {
        //normalization Each word is separated by only 1 space
         //normalize the dot and comma in the string s. There will be no spaces before
the dot or comma.
        //There must be a space after a dot or comma. The character after the dot
must be capitalized.
    }

    @Override
    public String F15_replace1(String s, char x) {
                // repalce all longest words in s by x
    }

    @Override
    public String F16_replace2(String s, char x) {
                // repalce all palidrom words by x
            }

    @Override
    public String F17_replace3(String s, char x) {
                // replace all shortest word in s by x

    }

    @Override
    public String F18_replace4(String s, char x) {
               //        replace the last longest word by x
      }

    @Override
    public int F19_find(String sub, String s) {
    // Returns the first position of substring sub in string s. if sub is not in s, return -1
    }
}
```

---------------- OUTPUT-------------



```
======MENU=========
1. Test count PCs by model
2. Test count PCs by price
3. Test count PCs by rate
4 Test sublist
5. Test find1()
6. Test find2()
7. Test find3()
8. Test find4()
9. Test remove all by model
10. Test remove all by model and rate
11. Test remove by model, rate and price
12. Test average price by model
13. Test countChar()
14 Test normalization()
15. Test replace1()
16. Test replace2()
17. Test replace3()
18. Test replace4()
19. Test find()
Enter TC(1-19): 5
INPUT:

Model   DEll    MAc     HP      ACer    ASus    ASus    HP      ACer    DEll    DEll    DEll
Rate    5.0     5.0     3.0     1.0     4.0     6.0     3.0     2.0     4.0     2.0     3.0
Pice    15000.0 15000.0 9000.0  5000.0  8000.0  15000.0 8000.0  10000.0 12000.0 11000.0 15000.0
OUTPUT:
(DEll,5.0,15000.0)
Test case 5 IS OK
BUILD SUCCESSFUL (total time: 6 seconds)
```