Präsenzaufgaben 6 11.11.2021

Die Lösung der Aufgaben wird am Ende der Übung von Ihnen vorgestellt.

ArrayLists der Klasse `Object`

Da eine `ArrayList<X>` alle Objekte der Klasse X und aller Unterklassen von X aufnehmen kann, kann eine `ArrayList<Object>` alle Objekte aufnehmen. Lässt man das Generic ganz weg, erhält man automatisch eine `ArrayList<Object>`.

- Leiten Sie von der Klasse `ArrayList` die Klasse `NewArrayList` ab und fügen Sie die drei folgenden Methoden hinzu:
  - `public void add(String x)`
  - `public void add(int x)`
  - `public void add(double x)`

  Die drei Methoden sollen den Übergabeparameter der Liste hinzufügen.
- Bei den Typen `int` und `double` soll der Übergabeparameter lediglich in die entsprechende Wrapper-Klasse eingebettet werden.
- Bei `String`-Übergabeparametern gibt es eine Besonderheit:
  - Lässt sich `x` in einen `int` -Wert umwandeln, soll er als `int` aufgenommen werden.
  - Lässt sich `x` in einen `double`-Wert umwandeln (aber nicht in einen `int`), soll er als `double` aufgenommen werden.
  - Ansonsten soll `x` als `String` aufgenommen werden.
- **Hinweis:** Folgendermaßen lässt sich prüfen, ob sich ein String zu einem `int` oder `double` umwandeln lässt:

```
String s = "-123456789012";  //Dieser String soll ueberprueft werden
Scanner sc = new Scanner(s); //Scanner auf den String aufmachen
sc.useLocale(Locale.US);     //Einstellen auf Dezimalpunkt (statt -komma)
System.out.println(sc.hasNextInt());    //Geht Umwandeln in int?
System.out.println(sc.hasNextDouble()); //Geht Umwandeln in double?
sc.close();
```

Erkennen des dynamischen Typs einer Variablen

Die Methode `get()` der `ArrayList<Object>` hat einen Rückgabewert vom Typ `Object`. Um zu sehen, welchen dynamischen Typ der Rückgabewert hat, gibt es 2 Möglichkeiten:

(1) Das Schlüsselwort `instanceof`:

```
if (x instanceof String) {
  //wird ausgefuehrt, wenn x vom dynamischen Typ String oder
  //einer Unterklasse ist.
}
```

(2) Die Methode `getClass()` der Klasse `Object`.

```
String s = x.getClass().getName();
//gibt den Namen der dynamischen Klasse von x zurück
```

- Fügen Sie der Klasse `NewArrayList` die folgenden Methoden hinzu:

  `public ArrayList<Integer> getIntegers()`
  `public ArrayList<Double> getDoubles()`

  Die Methoden sollen die Daten des jeweiligen Typs aus der Liste herausfiltern und in einer neu angelegten `ArrayList` zurückgeben.

Exceptions

Fügen Sie der Klasse `NewArrayList` auch eine Methode

```
public ArrayList<String> getStrings()
```

hinzu. Diese Methode wandelt Strings, Integers und Doubles in Strings und gibt sie in einer neuen Liste zurück. Sollten sich in der Liste andere Datentypen außer Strings, Integers und Doubles befinden, wird eine selbst zu schreibende `WrongElementException` geworfen. Fügen Sie der Exception-Klasse ein weiteres Attribut hinzu, welches vom Typ **`int`** ist und den Index des unpassenden Listeneintrags angibt. Erstellen Sie außerdem eine entsprechende getter-Methode `getIndex`. Die Fehlermeldung sollte das Format

```
Wrong data element in <x>
```

besitzen, wobei <x> der Index des fehlerhaften Elements ist.

XPoint

Sehen Sie sich die API der Java-Klasse `Point` an. Schreiben Sie anschließend eine Klasse `XPoint`, die von `Point` erbt und folgende Erweiterungen aufweist:

```
//Tauscht die Koordinaten des this-Objekts mit Punkt p
public void swap(Point p)

//Gibt zurueck, ob das this-Objekt auf der durch a und b beschriebenen
//Geraden liegt.
public boolean isOnLine(Point a, Point b)
```

Ein Punkt (x/y) lässt sich auch in Polarkoordinaten (r/φ) ausdrücken. Dabei ist r der Abstand zum Ursprung und φ der Winkel zur positiven x-Achse. Der Winkel von φ soll im Bogenmaß ausgegeben werden (360° sind im Bogenmaß 2π). Fügen Sie Ihrer Klasse die beiden folgenden Methoden zur Ermittlung der Polarkoordinaten eines Punkts hinzu:

```
//Gibt die Koordinate r des Punkts in Polarkoordinaten zurueck (entspricht
dem Abstand zum Koordinatenursprung).
public double getR()

//Gibt die Koordinate phi des Punkts in Polarkoordinaten (im Bogenmass)
zurück. Liegt der Punkt im Koordinatenursprung, wird 0 zurückgegeben.
public double getPhi()
```

Hinweise:

- Die Klasse `Point` speichert die Attribute x und y intern als **`int`**. In den setter-Methoden werden die **`double`**-Parameter in **`int`** gerundet.
- Benutzen Sie für die Berechnung von φ die Funktion `Math.atan2`. Sehen Sie sich die API hierzu an.
- Achten Sie darauf, dass Sie auch die Konstruktoren neu schreiben müssen. Testen Sie Ihre Klasse.