Introductory programming

Assignment 6
Hand-in date: Monday 05-12-2015 15:00h

6.1 Design a class named **Rectangle** to represent a rectangle. The class has to contain:

(a) Two **double** data fields named width and height that specify the width and height of the rectangle. The default values are **1** for both width and height.
(b) A no-arguments constructor that creates a default rectangle.
(c) Constructor that creates a rectangle with the specified width and height.
(d) A method named **getArea()** that returns the area of this rectangle.
(e) A method named **getPerimeter()** that returns the perimeter.

Write a test program that creates two **Rectangle** objects—one with width 4 and height 40 and the other with width 3.5 and height 35.9. Display the width, height, area, and perimeter of each rectangle in this order.

6.2 Write a program that creates a **Date** object, sets its elapsed time to **10000**, **100000**, **1000000**, **10000000**, **100000000**, **1000000000**, **10000000000**, and **100000000000** milliseconds, and displays the date and time using the **toString()** method, respectively.

Tip: You can use a loop to manage with the increasing milliseconds value.

6.3 Design a class named **StopWatch** . The class contains:

(a) Private data fields **startTime** and **endTime** with getter methods.
(b) A no-arguments constructor that initializes **startTime** with the current time.
(c) A method named **start()** that sets the **startTime** to the current time.
(d) A method named **stop()** that sets the **endTime** to the current time.
(e) A method named **getElapsedTime()** that returns the elapsed time for the stopwatch in milliseconds.

Write a test program that measures the execution time of sorting **100,000** random numbers using selection sort algorithm.

Note: Using Java's **Arrays.sort(some array)** method is not allowed in the exercise).

6.4 Design a class named **Account** that contains:

(a) A private **int** data field named **id** for the account (default **0**).
(b) A private **double** data field named **balance** for the account (default **0**).
(c) A private **double** data field named **annualInterestRate** that stores the current interest rate (default **0**). Assume all accounts have the same interest rate. If it is possible and suitable introduce this as static variable.

Assignment 6
Hand-in date: Monday 05-12-2015 15:00h

(d) A private **Date** data field named **dateCreated** that stores the date when the account was created.
(e) A no-argument constructor that creates a default account.
(f) A constructor that creates an account with the specified **id** and initial balance.
(g) The accessor and mutator methods for **id**, **balance**, and **annualInterestRate**.
(h) The accessor method for **dateCreated**.
(i) A method named **getMonthlyInterestRate()** that returns the monthly interest rate.
(j) A method named **getMonthlyInterest()** that returns the monthly interest.
(k) A method named **withdraw** that withdraws a specified amount from the account.
(l) A method named **deposit** that deposits a specified amount to the account.

Tip: The method **getMonthlyInterest()** is to return monthly interest, not the interest rate.

Monthly interest is **balance** * **monthlyInterestRate**. **monthlyInterestRate** is **annualInterestRate / 12** . Note that **annualInterestRate** is a percentage, e.g., like **4.5%**. You need to divide it by **100**.

Write a test program that creates an **Account** object with an account ID of **1122**, a balance of **$20,000**, and an annual interest rate of **4.5%.** Use the withdraw method to withdraw **$2,500**, use the deposit method to deposit **$3,000**, and print the balance, the monthly interest, and the date when this account was created.

6.5 (OPTIONAL) Suppose two line segments intersect. The two endpoints for the first line segment are **(x1, y1)** and **(x2, y2)** and for the second line segment are **(x3, y3)** and **(x4, y4).**

Write a program that prompts the user to enter these four endpoints and displays the intersecting point. The intersecting point can be found by solving a linear equation. Use the **LinearEquation** class in Programming Exercise **9.11** from the book to solve this equation. See Programming Exercise **3.25** for sample runs also.

Assignment 6
Hand-in date: Monday 05-12-2015 15:00h