XML

Ćwiczenie 1

Napisz metodę, która zliczy średnią arytmetyczną pensji pracowników:

```
+getAvgSalary():double
```

Wykorzystaj plik `staff.xml`

Ćwiczenie 2

Napisz metodę, która odczyta plik XMl, sparsuje go, a następnie zwróci listę imion i nazwisk. Wykorzystaj plik `staff.xml`:

```
+getNames():ArrayList<String>
```

W metodzie `main()` wypisz do konsoli wszystkie odczytane imiona i nazwiska.

Ćwiczenie 3

Napisz metodę, która sprawdzi i wypisze do konsoli maksymalny oraz minimalny ID użytkownika w pliku XML (`staff.xml`).

```
+getMinMaxStaffID():void
```

Ćwiczenie 4

Napisz metodę, która pozwoli na zapisanie pliku XML według następującej struktury:

```
ROOT
      STUDENTS
            STUDENT
                  NAME: John
                  LASTNAME: Simple
                  YEAR: 3
            STUDENT
                  NAME: Jane
                  LASTNAME: Doe
                  YEAR: 1
```

Plik zapisz w pakiecie `resources` pod nazwą `students.xml`

Ćwiczenie 5

Napisz metodę, która pozwoli na zapisanie pliku XML według następującej struktury:

```
ROOT
      COMPANY(name="Testowa")
            STARTS: 2008
            EMPLOYEES: 345
            VAT: 23
      COMPANY(name="Testowa 2")
            STARTS: 1979
            EMPLOYEES: 34345
            VAT: 40
      COMPANY(name="Testowa 3")
            STARTS: 1999
            EMPLOYEES: 5
            VAT: 8
```

Plik zapisz w pakiecie `resources` pod nazwą `companies.xml`

Ćwiczenie 6

Utwórz klasę Person, która będzie zawierała następująca `pola prywatne, gettery, settery` oraz `konstruktory`:

```
-name:String
-lastname:String
-company:String
-salary:double
-department:String
-yearOfBorn:int
```

Następnie, dodaj przykładowych 5 instancji tego obiektu do listy (`ArrayList`) i dostarcz metodę:

```
+savePeople(ArrayList<Person> people):void
```

która zapisze dane użytkowników do pliku XML pod nazwą `people.xml`

Ćwiczenie 7

Przygotuj pliki XML oraz napisz program, który zliczy statystyki dla kilku plików XML. Przykładowo metoda:

```
+countAverage(String filename, int a, int b):double
```

zliczy średnią arytmetyczną z plików `plik_[a].xml` … `plik_[b].xml`, gdzie a i b to liczby nieujemne całkowite i `a < b`.

Pliki powinny mieć strukturę:

```
<root>
      <date>2016-12-12</date>
      <values>
            <val>123</val>
            <val>523</val>
            <val>53</val>
            <val>134</val>
            <val>1563</val>
            <val>973</val>
            <val>7673</val>
      </values>
</root>
```

Dostarcz także metodę, do zapisu plików według przekazanych na liście wartości, np.

```
+saveXMLFile(String filename, ArrayList<Integer>);
```

Ćwiczenie 8

Posiadając plik `CSV` (`Comma Separated Values`) - czyli wartości oddzielone wybranym delimeterem o nagówku:

```
ID;Imię;Nazwisko;Kierunek;Rok;Średnia
```

np.:

```
ID;Imię;Nazwisko;Kierunek;Rok;Średnia
1;Paweł;Testowy;Architektura;1;4.25
```

Napisz metodę, która przekonwertuje taki plik z nagłówkiem na plik XML (bez nagłówka).

Ćwiczenie 9

Napisz program, który sprawdzi, czy w pliku XML (staff.xml) nie występują elementy o takim samym ID. Jeżeli występują, program powinien utworzyć kopię pliku staff_copy.xml, a w nim zapisać niepowtarzające się elementy. Przyjmij, że tylko pierwsze napotkane unikalne ID jest poprawne i je zapisuj do pliku kopii. Resztę duplikatów pomiń.

Ćwiczenie 10

Napisz program, który pozwoli na realizację odczytu pliku XML. Następnie dostarcz metodę, która do drugiego pliku XML przepisze tylko te tagi, które zawierają w nazwie literę 'a'.