APLICATIE CURS 4

Sinopsis

Se creeaza clasa Persoana dupa cum urmeaza:

- clasa contine doua informatii – nume si data de nastere a persoanei, cea din urma fiind memorata sub forma a 3 campuri separate: an, luna si zi
- nu trebuie sa se poata crea obiecte Persoana invalide. In acest scop, constructorul va fi marcat ca *private* si va fi creata o metoda publica statica *getInstance()*, care isi valideaza datele de intrare si intoarce fie *null* (in cazul in care datele de intrare sunt invalide) fie un nou obiect Persoana creat pe baza acelor date. Pentru validare se folosesc metode ajutatoare
- clasa va dispune de o metoda *toString()* care produce o varianta formatata a detaliilor persoanei, folosindu-se de asemenea de metode ajutatoare

Programul va citi de la tastatura un nume si o data de nastere si, daca acestea sunt valide, va crea cu ele un nou obiect *Persoana* si ii va afisa descrierea prin apelarea metodei *toString()*.

Etapa 1 - functionalitatea de baza

Structura clasei:

- campuri:
  - nume
  - an
  - luna
  - zi
- metode:
  - **private Persoana(String n, int a, int l, int z)** – constructor ce memoreaza valorile primite ca parametri direct in campuri, fara a le valida (validarile sunt facute in *getInstance()*). Fiind private, constructorul va putea fi apelat numai din interiorul clasei (il vom folosi in metoda getInstance()).
  - **public static Persoana getInstance(String n, String d)** → intoarce un nou obiect Persoana, sau null daca datele de intrare sunt invalide. Pentru validare se foloseste de metodele *valideazaNume()* si *valideazaData()* de mai jos:
    - **private static boolean valideazaNume(String n)** → verifica daca numele este nenul, are lungime mai mare ca 3 si contine un spatiu
    - **private static boolean valideazaData(String d)** → verifica daca data are 10 caractere, daca luna este intre 1 si 12 si daca ziua este intre 1 si 31 (fara a tine cont de luna). Se va presupune ca data este in formatul *ZZ-LL-AAAA*.
  - **public String toString()** → intoarce un string de forma *Domnul Popescu Ion s-a nascut in data de 23.03.1945* (prenumele si numele sunt inversate fata de cum au fost scrise in numele introdus de la tastatura). Pentru determinarea prefixului se va crea metoda urmatoare:
    - **private String apelativ()** → intoarce *Domnul* sau *Doamna* in functie de ultima litera din prenume. Convenim ca vocalele indica doamne si consoanele domni. Vom

presupune ca prenumele este primul dintre cele doua cuvinte care formeaza numele complet
- **main()** → se citesc de la tastatura numele si data nasterii si cu ele se creeaza un nou obiect Persoana. Se afiseaza fie descrierea persoanei in cauza (rezultatul metodei *toString()*), fie o eroare in cazul in care datele de intrare sunt invalide

Etapa 2 - rafinare

- validarile numelui si datei vor fi facute mai in detaliu:
  - nume - metoda **valideazaNume()** va verifica daca numele este format din doua cuvinte separate printr-un spatiu (*prenume nume*) dupa cum urmeaza:
    - lungimea numelui trebuie sa fie cel putin 3
    - numele trebuie sa contina un singur spatiu, care nu se poate afla nici pe prima nici pe ultima pozitie
    - cele doua cuvinte ce bordeaza spatiul sunt formate exclusiv din litere mici sau mari
  - data - metoda **valideazaData()** va verifica daca data este valida si in formatul ZZ-LL-AAAA dupa cum urmeaza:
    - sa existe caracterul - (minus) pe pozitiile 2 si 5
    - restul de caractere sa fie cifre
    - anul sa fie in trecut
    - luna sa fie intre 1 si 12
    - ziua sa fie intre 1 si maximul lunii in cauza
- la afisarea detaliilor persoanei in metoda *toString():*
  - numele si prenumele vor fi formatate sa inceapa cu litera mare si sa continue cu litere mici, indiferent cum au fost introduse de user de la tastatura
  - sirul afisat va fi de forma *Domnul Popescu Ion s-a nascut in data de 23 martie 1945* (va fi afisat numele lunii, nu numarul acesteia)

In scopul acestor validari se creeaza urmatoarele clase utilitare suplimentare:
- **StringUtils** - clasa va contine exclusiv metode statice pentru procesare de siruri de caractere, dupa cum urmeaza:
  - **public static boolean onlyLetters(String s)** → verifica daca sirul primit ca parametru este format numai din litere mici sau mari. Folosim in acest scop *Character.isLetter(char)* aplicat fiecarui caracter din sirul s
  - **public static boolean onlyDigits(String s)** → verifica daca sirul primit ca parametru este format numai din cifre. Folosim in acest scop *Character.isDigit(char)*
  - **public String upperFirst(String s)** → face prima litera mare si restul mici (utilizabila pentru formatare nume si prenume)
- **DateUtils** - clasa va contine de asemenea exclusiv metode statice, dupa cum urmeaza:
  - **public static int zilePerLuna(int luna, int an)** - intoarce numarul de zile ale lunii in cauza in acel an
  - **public static String numeLuna(int luna)** - intoarce numele romanesc al lunii in cauza in functie de numarul ei