**Projekat 2 : Neuronska mreža**

Napisati klasu **NeuronskaMreza** koja sadrži polja: *brSkrivenihSlojeva* (celobrojnog tipa) i *brNeuronaPoSloju* (celobrojnog tipa). Ova klasa opisuje neuronsku mrežu koja u sebi uvek ima 1 ulaz i 1 izlaz, a navedena polja definišu koliko dodatnih neurona mreža ima. Klasa **NeuronskaMreza** sadrži:

* konstruktor sa parametrima – po jedan parametar za postavljanje svakog od polja, ukoliko je vrednost nekog od parametara negativan broj postaviti to polje na vrednost 0;
* metodu void dodajSkriveniSloj() – uvećava broj skrivenih slojeva za jedan i ispisuje na ekran taj broj;
* metodu void dodajNeuron() – uvećava broj neurona po skrivenom sloju za jedan i ispisuje na ekran taj broj;
* metodu double izračunajUspeh() const – računa i vraća trenutnu tačnost neuronske mreže (kao realan broj, voditi računa o tome prillikom pisanja formule) po formuli:
* metodu TipMreze getTipMreze() const – ukoliko mreža sadrži broj neurona veći ili jednak 1000 (uključujući i ulazni i izlazni neuron), metoda vraća vrednost DUBOKA (nabrojivog tipa **TipMreze**), u suprotnom vraća vrednost OBICNA (nabrojivog tipa **TipMreze**).

Napisati klasu **Test** koja sadrži polja: *nm* (tipa **NeuronskaMreža**), *zeljeniProcenatUspesnosti* (realnog tipa), *brOptimizacija* (celobrojnog tipa), *stanje* (nabrojivog tipa **StanjeTesta**). **Test** je opisan sledećim dijagramom:

procesiraj()

procesiraj()

potrebnoOptimizovati ()

procesiraj()

procesiraj()

procesiraj()

optimizuj(TipOptimizacije)

Klasa **Test** sadrži:

* konstruktor sa parametrima – samo jedan parametar za postavljanje željenog procenta uspešnosti (ukoliko je prosleđen broj koji nije u intervalu [0, 1] postaviti vrednost polja na 0.85), ostale vrednosti polja „hard kodirati“: *stanje* na START, *brOptimizacija* na 0, a *nm* na neuronsku mrežu sa 3 skrivena sloja sa 200 neurona po sloju;
* metodu bool procesiraj() – videti dijagram. Ukoliko je test u stanju START ili OPTIMIZACIJA, i ukoliko je polje *zeljeniProcenatUspesnosti* veće od trenutno izračunatog uspeha (tj. tačnosti) neuronske mreže koju test koristi, test prelazi u stanje PROCESIRANJE i uspešno se završava. Ukoliko je test u stanju START ili OPTIMIZACIJA, i ukoliko je polje *zeljeniProcenatUspesnosti* manje ili jednako od trenutno izračunatog uspeha (tj. tačnosti) neuronske mreže koju test koristi, test prelazi u stanje KRAJ, ispisuje rezultate testa (broj optimizacija, željenu uspešnost, postignut uspešnost (tačnost) i poruku „Test uspesno zavrsen!“, i uspešno se završava. U preostalim slučajevima metoda se završava neuspešno;
* metodu bool potrebnoOptimizovati() – videti dijagram;
* metodu bool optimizuj(TipOptimizacije) – videti dijagram. Metoda se može pozvati samo u stanju OPTIMIZACIJA (u tom slučaju izvršavanje se smatra uspešnim, u suprotnom se smatra neuspešnim). Ukoliko je taj uslov ispunjen, i ukoliko je vrednost parametra DODAJ\_SLOJ (nabrojivog tipa **TipOptimizacije**) dodaje se skriveni sloj neuronskoj mreži koju test koristi. Ukoliko je taj uslov ispunjen, i ukoliko je vrednost parametra DODAJ\_NEURON (nabrojivog tipa **TipOptimizacije**) dodaje se jedan neuron u skrivene slojeve neuronske mreži koju test koristi;
* metodu double izračunajUspeh() const – vraća trenutnu tačnost neuronske mreže koju test koristi;
* metodu TipMreze getTipMreze() const – vraća tip neuronske mreže koju test koristi;
* metodu StanjeTesta getStanje() const – vraća stanje testa;

Realizovati i slobodnu funkciju void ispisiTest(const **Test**&), koja će ispisivati trenutno stanje testa, tip neuronske mreže koju test koristi, kao i trenutni uspeh (tačnost) neuronske mreže koju test koristi.

**Napomena :**

Napraviti funkciju meni koja omogućava pozivanje svih metoda nad jednim objektom.