S1.

Să se creeze o clasă cu numele *Punct*. Pentru ea se cer îndeplinite următoarele:

- definirea unor variabile cât mai relevante (*x* si *y* de tip double);
- constructorul clasei, forma fără inițializare și cea cu inițializare; **(1p.)**
- funcția *afisare( )*; **(1p.)**
- definirea a cel puțin două funcții de care să beneficieze obiectele clasei (exemple: translație, oglindire față de axa verticală (mirror), oglindire față de axa orizontală etc.); **(2p.)**
- definirea unei clase derivate (*Cerc*) din clasa de mai sus; **(1p.)**
- constructorul pentru clasa derivată, cu transmiterea datelor pentru inițializare constructorului clasei de bază; **(1p.)**
- supraîncărcarea funcțiilor din clasa de bază; **(1p.)**
- folosirea celor două clase într-un program în care să se exemplifice crearea unor obiecte din clasele respective și folosirea funcțiilor membre, definite pentru ele. **(2p.)**

S2.

Să se implementeze o clasă cu numele *Punct*. Pentru aceasta trebuie îndeplinite următoarele cerințe:

- definirea unor variabile cât mai relevante (*x* si *y* de tip double);
- constructorul clasei, forma cu inițializare și cea fără; **(1p.)**
-funcția *afisare( )*; **(1p.)**
-definirea a cel puțin două funcții membre de care să beneficieze obiectele clasei (exemple: translație, oglindire față de axa orizontală (mirror), oglindire față de axa orizontală etc.); **(2p.)**
- definirea unei clase derivate (*Triunghi*) din clasa de mai sus; **(1p.)**
- constructorul pentru clasa derivată, forma cu inițializare și cea fără inițializare; **(1p.)**
- supraîncărcarea funcțiilor din clasa de bază; **(1p.)**
- folosirea celor două clase într-un program în care se vor crea obiecte din cele două clase și se vor folosi toate funcțiile membre. **(2p.)**

S3.

Să se creeze o clasă cu numele *circuit_RLC* pentru care se cer următoarele:

- definirea unor variabile cât mai relevante (V1, I1 si Z1 de tip complex și f1, R, L, C de tip double);
- constructorul clasei, forma fără inițializare și cea cu inițializarea V1, f1, R, L, C; **(1p.)**
- funcția *afisare ( )*; **(1p.)**
- definirea a cel puțin două funcții, în afară de *afisare(),* de care să beneficieze obiectele clasei sau clasele derivate (în această categorie intră și funcțiile pentru inițializarea sau modificarea tensiunii de alimentare V1 sau a parametrilor f1, R, L, C); **(1p.)**
- definirea unei clase derivate (*serie_RLC*) din clasa de mai sus; **(2p.)**
- constructorul pentru clasa derivată; **(1p.)**
- supraîncărcarea funcțiilor din clasa de bază; **(1p.)**

În funcția *main( )* se vor crea obiecte din clasa *serie_RLC* și se vor folosi toate funcțiile ei membre. **(2p.)**

S4.

Să se creeze o clasă cu numele *circuit_RLC* pentru care se cer:

- definirea unor variabile cât mai relevante (V1, I1 si Z1 de tip complex și f1, R, L, C de tip double);
- constructorul clasei, forma fără inițializare și cea cu inițializarea V1, f1, R, L, C; **(1p.)**
- funcția *afisare( )*; **(1p.)**
- definirea a cel puțin două funcții în afară de *afisare( )* de care să beneficieze obiectele clasei sau clasele derivate (în această categorie intră și funcțiile pentru inițializarea sau modificarea tensiunii de alimentare V1 sau a parametrilor f1, R, L, C); **(1p.)**
- definirea unei clase derivate (*paralel_RLC*) din clasa de mai sus; **(2p.)**
- constructorul pentru clasa derivată; **(1p.)**
- supraîncărcarea funcțiilor din clasa de bază; **(1p.)**
- folosirea clasei derivate în *main( ),* prin care să se exemplifice crearea unor obiecte si folosirea tuturor funcțiilor membre. **(2p.)**

S5.

Să se creeze o clasă cu numele *Raţional*. Pentru această clasă trebuie îndeplinite următoarele cerinţe:

- definirea unor variabile cât mai relevante (*m* si *n* de tip întreg);
- constructorul clasei, forma cu iniţializare şi cea fără; **(1p.)**
- funcţia *afisare( )*; **(1p.)**
- supraîncărcarea operatorului +; **(1p.)**
- supraîncărcarea operatorului -; **(1p.)**
- supraîncărcarea operatorului *; **(1p.)**
- supraîncărcarea operatorului /; **(1p.)**
- supraîncărcarea operatorului +=; **(1p.)**

În funcţia *main( )* se vor crea obiecte din clasa *Raţional* şi se vor folosi funcţiile membre şi operatorii care au fost supraîncărcaţi. **(2p.)**

S6.

Să se implementeze o clasă cu numele *Complex* pentru care se cer îndeplinite următoarele:

- definirea unor variabile cât mai relevante pentru clasă (*re* si *im* de tip double);
- constructorul clasei, cu și fără inițializare; **(1p.)**
- funcția *afisare( )*; **(1p.)**
- două funcții, una care returnează partea reală, cealaltă partea imaginară; **(1p.)**
- supraîncărcarea operatorului +=; **(1p.)**
- supraîncărcarea operatorului +; **(1p.)**
- supraîncărcarea operatorului -; **(1p.)**
- supraîncărcarea operatorului *; **(1p.)**
- folosirea clasei *Complex* în funcția *main( )*. Se vor crea obiecte din clasă și se vor folosi toate funcțiile membre și toți operatorii supraîncărcați. **(2p.)**

S7.

Să se creeze o clasă cu numele *Student*. Pentru această clasă trebuie îndeplinite următoarele cerințe:

- definirea unor variabile cât mai relevante pentru clasa de mai sus (*char nume[25]*, *char prenume[30]* și *int nota[5]*);
- constructorul clasei; **(1p.)**
- funcția prin care se citesc numele, prenumele și notele obținute la o sesiune de examene de către student (fiecare student va avea cinci note, conform variabilei membre *int nota [5]*); **(2p.)**
- funcția *afișare( ),* care trebuie să afișeze pe ecran numele și prenume studentului precum și notele și media lor (în caz că o notă este mai mică decât 5 se va afișa lângă notă mesajul „nepromovat"); **(2p.)**
- funcția pentru calculul mediei de promovare (media celor cinci note; dacă două note din cele cinci sunt sub 5 se afișează mesajul: "Condiție de calcul a mediei neîndeplinită!") ; **(2p.)**
- folosirea clasei în *main( )*. Se va crea un **tablou de obiecte (o matrice unidimensională)** din clasa *Student* si se vor folosi toate funcțiile membre ale clasei. **(2p.)**

S8.

Să se implementeze o clasă cu numele *Triunghi*. Se cer:

- definirea unor variabile cât mai relevante pentru clasa de mai sus (trei variabile de tip *Punct*, cu numele *p1, p2, p3*, sau o matrice unidimensională cu trei elemente, *Punct p[3],* pentru vârfurile triunghiului - unde *Punct* este o structură cu două variabile de tip double, x și y);
- constructorul clasei, cu și fără inițializare; **(1p.)**
- o funcție prin care se citesc coordonatele vârfurilor triunghiului de la tastatură; **(1p.)**
- funcția *verificare_existenta_triunghi()* (cele trei puncte care reprezintă vârfurile triunghiului să nu fie pe aceeași dreaptă); **(2p.)**
- funcția *afisare( )*; **(1p.)**
- funcția pentru calculul lungimilor laturilor si al perimetrului; **(1p.)**
- funcția pentru calculul ariei triunghiului; **(1p.)**
- folosirea clasei în *main( )*. Se vor crea obiecte din clasa *Triunghi* și se vor apela toate funcțiile membre ale clasei. **(2p.)**