SUBIECTUL NR. 1

1. Concepeţi o clasă adecvată pentru implementarea **structurii dinamice de date de tip listă liniară simplu înlănţuită** cu informații de tip întreg**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) verificarea listei vide (inline)
   e) căutarea unui nod de cheie dată
   f) adăugarea unui nod nou la începutul listei
   g) adăugarea unui nod nou la sfârşitul listei
   h) ștergerea nodurilor cu o anumită informație
   i) operatorul += pentru concatenarea a două liste
   j) operatori supraîncărcaţi pentru citire/scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - j şi cu opţiunea *Ieşire*. Programul de test va utiliza liste citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 2

1. Concepeţi o clasă adecvată pentru implementarea **structurii dinamice de date de tip stivă (LIFO)** cu informații de tip întreg.
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) verificarea stivei vide (inline)
   e) PUSH (adăugarea unui nod nou)
   f) POP (eliminarea vârfului stivei, cu returnarea nodului eliminat)
   g) TOP (returnează informația din vârful stivei)
   h) Split (împărțirea unei stive în alte 2 stive, după al k-lea nod)
   i) operatorul += pentru concatenarea a două stive
   j) operatori supraîncărcaţi pentru citire/scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - j şi cu opţiunea *Ieşire*. Programul de test va utiliza liste citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 3

1. Concepeţi o clasă adecvată pentru implementarea **structurii dinamice de date de tip coadă (FIFO)** cu informații de tip întreg.
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) verificarea coadă vidă (inline)
   e) BACK (adăugarea unui nod nou la sfârșit)
   f) FRONT (eliminarea primului nod, cu returnarea nodului eliminat)
   g) FIRST (returnează informația din primul nod)
   h) Split (împărțirea cozii în alte 2 stive, după al k-lea nod)
   i) operatorul –= pentru eliminarea nodurilor cu o cheie dată
   j) operatori supraîncărcaţi pentru citire/scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - j şi cu opţiunea *Ieşire*. Programul de test va utiliza liste citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 4

1. Concepeţi o clasă adecvată pentru implementarea **structurii dinamice de date de tip listă circulară simplu înlănțuită** cu informații de tip întreg.
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) verificarea listei vide (inline)
   e) căutarea unui nod de cheie dată cu returnarea primei adrese a acestuia sau NULL
   f) adăugarea unui nod nou la începutul listei
   g) adăugarea unui nod nou la sfârşitul listei
   h) ștergerea nodurilor din k în k cu afișarea acestora în ordinea eliminării
   i) operatorul += pentru concatenarea a două liste
   j) operatori supraîncărcaţi pentru citire/scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - j şi cu opţiunea *Ieşire*. Programul de test va utiliza liste citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 5

1. Concepeţi o clasă adecvată pentru implementarea **structurii dinamice de date de tip listă liniară dublu înlănţuită** cu informații de tip întreg**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) verificarea listei vide (inline)
   e) căutarea unui nod de cheie dată
   f) adăugarea unui nod nou la începutul listei
   g) adăugarea unui nod nou la sfârşitul listei
   h) ștergerea nodurilor cu o anumită informație
   i) operatorul += pentru concatenarea a două liste
   j) operatori supraîncărcaţi pentru citire/scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - j şi cu opţiunea *Ieşire*. Programul de test va utiliza liste citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 6

1. Concepeţi o clasă adecvată pentru implementarea **structurii dinamice de date de tip listă liniară simplu înlănţuită ordonată crescător** cu informații de tip întreg**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) verificarea listei vide (inline)
   e) căutarea unui nod de cheie dată
   f) adăugarea unui nod nou cu păstrarea proprietății
   g) ștergerea nodurilor cu o anumită informație
   h) operatorul + pentru interclasarea a două liste
   i) operatori supraîncărcaţi pentru citire/scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - j şi cu opţiunea *Ieşire*. Programul de test va utiliza liste citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 7

1. Concepeţi o clasă adecvată pentru implementarea **structurii dinamice de date de tip matrice rară.** O matrice A(n,m) se numeşte *rară* dacă cel puţin 70% dintre elementele ei sunt nule**.** Matricele rare se memorează în formă condensată (linie, coloană şi valare nenulă), în ordinea lexicografică a perechilor de indici.
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) nr_linii, nr_coloane (inline + returnează numărul de linii / colane)
   e) maximul din matrice
   f) minimul din matrice
   g) elementul de pe linia L, coloana C
   h) operatorul + pentru adunarea a 2 matrice rare
   i) operatori supraîncărcaţi pentru citire/scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - i şi cu opţiunea *Ieşire*. Programul de test va utiliza matrice în format condensat citite atât din fişier cât şi de la tastatură.

SUBIECTUL NR. 8

1. Concepeţi o clasă adecvată pentru implementarea **structurii dinamice de date de tip polinom rar.** Un polinom de grad n, P(n,X) se numeşte *rar* dacă cel puţin 70% dintre coeficienţi sunt nuli**.** Polinoamele rare se memorează în formă condensată (indice şi valoare coeficient nenul), în ordinea reprezentării canonice.
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) gradul polinomului (inline)
   e) valoarea polinomului în punctul X
   f) operatorul * pentru înmulţirea cu scalar
   g) operatorul + pentru adunarea a 2 polinoame rare
   h) operatori supraîncărcaţi pentru citire/scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e –h şi cu opţiunea *Ieşire*. Programul de test va utiliza polinoame citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 9

1. Concepeţi o clasă adecvată pentru implementarea **grafurilor neorientate 1** reprezentate prin matrice de adiacenţă şi date prin numărul de noduri şi lista de muchii**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) returnarea numărului de noduri (inline)
   e) verificarea adiacenței a două noduri
   f) parcurgerea DFS
   g) parcurgerea BFS
   h) componente conexe (metoda va returna numărul lor și le va afișa)
   i) operatorul < pentru verificarea proprietății de graf parțial (A<B <=> A este graf parțial al lui B)
   j) operatori supraîncărcaţi pentru citire/scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - j şi cu opţiunea *Ieşire*. Programul de test va utiliza grafuri citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 10

1. Concepeţi o clasă adecvată pentru implementarea **grafurilor neorientate 2** reprezentate prin liste de adiacenţă şi date prin numărul de noduri şi lista de muchii**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) returnarea numărului de noduri (inline)
   e) număr vârfuri izolate
   f) verificare graf complet
   g) determinare lanţ de lungime minimă de la x la y, cu returnarea lungimii acestuia și a lanţului
   h) verificare graf aciclic
   i) operatori supraîncărcaţi pentru citire/scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - i şi cu opţiunea *Ieşire*. Programul de test va utiliza grafuri citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 11

1. Concepeţi o clasă adecvată pentru implementarea **grafurilor neorientate 3** reprezentate prin matrice de adiacenţă şi date prin numărul de noduri şi lista de muchii**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) returnarea numărului de noduri (inline)
   e) gradul unui nod
   f) verificare conexitate
   g) verificare ciclu eulerian (dată o succesiune de noduri, săse verifice dacă este ciclu eulerian)
   h) determinare ciclu eulerian
   i) operatori supraîncărcaţi pentru citire (cu validare graf eulerian)/scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - i şi cu opţiunea *Ieşire*. Programul de test va utiliza grafuri citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 12

1. Concepeţi o clasă adecvată pentru implementarea **grafurilor neorientate 4** reprezentate prin matrice de adiacenţă şi date prin numărul de noduri şi lista de muchii**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) returnarea numărului de noduri (inline)
   e) gradul unui nod
   f) verificare graf bipartit
   g) verificare ciclu hamiltonian (dată o succesiune de noduri, să se verifice dacă este ciclu hamiltonian)
   h) determinare ciclu hamiltonian (returneaza ciclul hamiltonian și existența acestuia)
   i) operatori supraîncărcaţi pentru citire /scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - i şi cu opţiunea *Ieşire*. Programul de test va utiliza grafuri citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 13

1. Concepeţi o clasă adecvată pentru implementarea **grafurilor neorientate 5** reprezentate prin matrice de adiacenţă şi date prin numărul de noduri şi lista de muchii**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) returnarea numărului de noduri (inline)
   e) afisare lista adiacență pentru un nod dat
   f) determinare matricea lanțurilor
   g) operatorul – pentru eliminarea unui nod și returnarea subgrafului obținut
   h) operatorul ~ pentru determinare graf complementar
   i) operatori supraîncărcaţi pentru citire /scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - i şi cu opţiunea *Ieşire*. Programul de test va utiliza grafuri citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 14

1. Concepeţi o clasă adecvată pentru implementarea **grafurilor orientate 1** reprezentate prin matrice de adiacenţă şi date prin numărul de noduri şi lista de arce**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) returnarea numărului de noduri (inline)
   e) afisare lista adiacență pentru un nod dat
   f) determinare grad interior
   g) determinare grad exterior
   h) parcurgere DFS din nodul k
   i) parcurgere BFS din nodul k
   j) operatori supraîncărcaţi pentru citire /scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - j şi cu opţiunea *Ieşire*. Programul de test va utiliza grafuri citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 15

1. Concepeţi o clasă adecvată pentru implementarea **grafurilor orientate 2** reprezentate prin matrice de adiacenţă şi date prin numărul de noduri şi lista de arce**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) returnarea numărului de noduri (inline)
   e) verificare graf turneu
   f) matricea drumurilor
   g) verificare tare conexitate
   h) determinare componente tari conexe
   i) operatorul < (verifica dacă A este graf parţial al lui B)
   j) operatori supraîncărcaţi pentru citire /scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - j şi cu opţiunea *Ieşire*. Programul de test va utiliza grafuri citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 16

1. Concepeţi o clasă adecvată pentru implementarea **arbori binari** reprezentate prin număr de noduri şi vectori de fii st, dr**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) Parcurgere in preordine
   e) Parcurgere in inordine
   f) Parcurgere in postordine
   g) Determinare nivel pentru un nod dat
   h) Determinare înălţime arbore
   i) Afişare frunze
   j) Afişare noduri de pe un anumit nivel
   k) operatori supraîncărcaţi pentru citire /scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e – k şi cu opţiunea *Ieşire*. Programul de test va utiliza arbori citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 17

1. Concepeţi o clasă adecvată pentru implementarea **arbori cu rădăcină** reprezentate prin număr de noduri, rădăcină şi vectori de taţi**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) determinare nivel pentru un nod dat
   e) determinare înălţime arbore
   f) afişare frunze
   g) afişare noduri fraţi pentru un anumit nod
   h) operatorul == care verifică egalitatea a doi arbori
   i) operatori supraîncărcaţi pentru citire /scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - j şi cu opţiunea *Ieşire*. Programul de test va utiliza arbori citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 18

1. Concepeţi o clasă adecvată pentru implementarea **numere mari** reprezentate prin vectori de cifre**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) verifică paritatea (inline)
   e) operator de comparare a două numere mari (returnează -1, 0, 1 după cum a<b, a==b, a>b)
   f) operator de adunare +
   g) operator de scădere -
   h) operator de înmulțire * cu o cifră
   i) operatori supraîncărcaţi pentru citire /scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - j şi cu opţiunea *Ieşire*. Programul de test va utiliza numere mari citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 19

1. Concepeţi o clasă adecvată pentru implementarea **mulțimilor de numere întregi** reprezentate prin vectori de elemente ordonate strict crescător**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) returnează numărul de elemente ale mulțimii (inline)
   e) operator de adunare + pentru reuniunea a două mulțimi
   f) operator de adunare - pentru diferența a două mulțimi
   g) operator de înmulțire * pentru intersecția a două mulțimi
   h) apartenența (verifică dacă un număr face parte din mulțime)
   i) operator < pentru incluziune
   j) operator == pentru egalitatea a două mulţimi
   k) operatori supraîncărcaţi pentru citire /scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - j şi cu opţiunea *Ieşire*. Programul de test va utiliza mulțimile citite atât de la tastatură cât şi din fişier.

SUBIECTUL NR. 20

1. Concepeţi o clasă adecvată pentru implementarea **mulțimilor de numere naturale** (cu cel mult 4 cifre) reprezentate prin vectori caracteristici**.**
   Clasa va conţine metodele:
   a) constuctor
   b) constructor de copiere
   c) destructor
   d) returnează numărul de elemente ale mulțimii (inline)
   e) operator de adunare + pentru reuniunea a două mulțimi
   f) operator de adunare - pentru diferența a două mulțimi
   g) operator de înmulțire * pentru intersecția a două mulțimi
   h) apartenența (verifică dacă un număr face parte din mulțime)
   i) operator < pentru incluziune
   j) operator == pentru egalitatea a două mulţimi
   k) operatori supraîncărcaţi pentru citire /scriere

2. Construiţi o aplicaţie C++ care să conţină clasa ca fişier Header (.h), definitiile metodelor clasei in fisier sursă (.cpp) si fişierul de testare a clasei (.cpp). Fişierul de testare (main) va permite selectarea operațiilor prin intermediul unui meniu cu opţiunile corespunzătoare punctelor e - j şi cu opţiunea *Ieşire*. Programul de test va utiliza mulțimile citite atât de la tastatură cât şi din fişier.

Documentație aplicație

Documentaţia proiectului va fi organizată după structura următoare, listată în format A4 şi îndosariată.

1. Prima pagina va contine centrat titlul proiectului: Clasa **……., Elev:** numele, clasa, an scolar 2014-2015, sem II**.**
2. Descrierea clasei (cu comentarii explicative la date şi metode)
3. Implementarea clasei
4. Aplicaţii: prezentarea aplicaţiilor selectate şi evidenţierea modului de adaptare a implementării cu ajutorul clasei(sursele comentate şi listate)
5. Aplicaţia arhivata cu numele vostru si trimisa pe mail la mirela_tibu@yahoo.com cornelia_ivasc@yahoo.com
6. Paginile proiectului vor fi numerotate si vor avea un header cu Titlul proiectului, numele si clasa elevului (cu excepţia primei pagini)

BAREM: 100 puncte

10 p – structura clasei (alegerea adecvată a datelor şi metodelor şi a modului lor de implementare)
20 p – implementarea corectă a funcţiilor membru
10 p – implemetarea meniului
20 p – aplicaţia corect implementată
10 p – funcţionarea corectă a aplicaţiei
10 p – dosarul de prezentare în formatul precizat
10 p – prezentarea/susţinerea adecvată lucrării
10 p – oficiu

TERMEN DE REALIZARE: până pe 20 mai 2015

CLASA a XI–a A

Nr. Crt.	Nume si prenume	Nr. subiect	S1	S2	S3
1.					
2.					
3.					
4.					
5.					
6.					
7.					
8.					
9.					
10.					
11.					
12.					
13.					
14.					
15.					
16.					