Programare Orientată Obiect
= Tema 1 =

Scop general:

- recapitularea cunoștințelor de programare în limbaj C;
- utilizarea mediului de dezvoltare Visual Studio;

Scop specific:

- deprinderea cu particularitățile lucrului în mediul de dezvoltare Visual C++: lucrul cu proiecte, utilizare editor/compilator, depanarea aplicațiilor prin execuție pas cu pas, inspectarea variabilelor;
- deprinderea cu utilizarea MSDN pentru regăsirea informațiilor utile;
- lucrul cu vectori și matrici de date; structurarea codului cu funcții, organizarea codului în fișiere, structuri de date;
- lucrul cu șiruri de caractere; memorarea șirurilor de caractere cu terminator '\0'; structuri de date;

Documentație specifică:

- carte C/C++; recomandarea noastră: H. Schildt, The complet reference C++, 3rd edition, ed. McGrow Hill, 1999, ISBN 0-07-882476-1;
- Microsoft MSDN 6, 2CD sau web site;
- lucrări de laborator la disciplina PCLP 1 & 2 și POO, Fac. ETTI Iași;

Cerințe:

- tema conține două probleme. Enunțul fiecărei probleme permite o rezolvare cu o complexitate ce crește gradual (precizând o soluție minimală); enunțul și modalitatea de rezolvare a fost discutat în săptămâna II;
- pentru fiecare problemă va fi completată o fișă de verificare dovedind cunoașterea soluției;
- notarea este proporțională cu complexitatea și corectitudinea soluției furnizate pe baza informației din fișa de verificare;
- soluția realizată va putea fi utilizată ulterior în cadrul proiectului la disciplina POO;
- tema se rezolvă personal prin verificarea codului sursă pe calculator pentru date de intrare specifice;
- tema se predă semnată, scris de mână sau tipărit față-verso, perforat și îndosariat în dosar cu șină. **Tema conține tot codul sursă indicând și numele fișierelor și este însoțită de fișa de verificare.** Tema se poate verifica prin întrebări la predare sau până la definitivarea notei la disciplina POO; Temele se vor păstra intr-un dosar;
- termen de predare: **în săptămâna V,** la începutul laboratorului;

Problema I

Cerințe minimale:

Pentru un vector cu elemente întregi de dimensiune n (n<=100) ale cărui elemente se citesc de la consolă și apoi se afișează cate 5 pe linie, se cere:

a. calculul sumei și produsului elementelor sale;
b. determinarea elementului maxim, a elementului minim și a pozițiilor pe care se află acestea;
c. ordonarea elementelor sale crescător/ descrescător.

Pentru 2 vectori X și Y cu n elemente să se calculeze și să se afișeze:

d. vectorul sumă S= X + Y și vectorul produs P = X .* Y;
e. produsul scalar al celor 2 vectori X si Y;
f. X*Y (interpretând X ca fiind vector coloană (n x 1) și Y ca vector linie (1x n)).

Pentru o matrice de dimensiuni mxn (m,n preluați de la consolă) să se calculeze:

a. suma și produsul elementelor matricii;
b. elementul maxim, elementul minim și pozițiile acestora;
c. numărul elementelor negative, numărul celor nule și numărul celor pozitive;

Pentru două matrici A,B cu dimensiuni preluate de la consola:

d. să calculeze suma elementelor de pe diagonala principală a unei matrici pătratice;
e. să se calculeze și afișeze matricea sumă C=A+B (cele trei matrici au aceleași dimensiuni mxn,);
f. să se calculeze și afișeze matricea produs C=A(mxn)*B(nxp).

Cerințe în completare (1):

Codul să se organizeze cu funcții pentru citirea respectiv afișarea datelor. Se vor implementa funcții pentru operațiile solicitate și se vor apela corespunzător.

Cerințe în completare (2):

Se vor defini structuri de date pentru Vector și respectiv pentru Matrice. Funcțiile de citire/calcul și afișare se vor proiecta corespunzător.

Indicație: în structura de date se pot reuni: datele, dimensiunea lor, un șir de caractere cu numele variabilei, un indicator pentru date valide.

Cerințe în completare (3):

Datele din structurile de date pentru Vector și Matrice se vor aloca dinamic. Funcțiile de citire/calcul și afișare se vor implementa corespunzător. Vor fi proiectate funcții pentru inițializarea structurilor de date, pentru alocare și pentru eliberarea memoriei.

Indicație: în structura de date se pot reuni: datele alocate dinamic (pointer la date), dimensiunea lor, un șir de caractere cu numele variabilei, un indicator pentru memorie alocată, un indicator pentru date valide.

Un exemplu de alocare de memorie pentu matrici este în lucrările de laborator pentru POO, de exemplu:

Laborator 2: Alocarea dinamică a memoriei (02_Lab02.pdf)

Problema I (fișa de verificare)

Descriere implementare:

Soluția implementează:

Cerința	autoevaluare	evaluare
numai cerința minimală		
funcțiile pentru citirea/afișarea vectorilor în fișierul …… prin funcțiile ……		
funcțiile pentru citirea/afișarea de matrici în fișierul …… prin funcțiile …….		
funcțiile pentru calcul cu vectori în fișierul ……..…… prin funcțiile …….		
funcțiile pentru calcul cu matrici în fișierul …… prin funcțiile ……		
structurile de date pentru vectori cu numele …… declarate în fișierul ……		
funcțiile referitoare la vectori utilizează structuri de date		
implementarea structurii de date pentru vectori și a funcțiilor ..... …… pentru vectori sunt cu alocare dinamică		
există funcții pentru alocarea/eliberarea memoriei în cazul vectorilor ……		
structurile de date pentru matrici cu numele …… declarate în fișierul ……		
funcțiile referitoare la matrici utilizează structuri de date		
implementarea structurii de date pentru matrici și a funcțiilor pentru matrici sunt cu alocare dinamică		
exista funcții pentru alocarea/eliberarea memoriei în cazul matricilor ……		

Implementarea conține:

operații de intrare / ieșire cu **scanf** / **printf**		
operații de intrare / ieșire cu **cin** / **cout** și << / >>		
fișiere header.h cu protecție la dublă includere		
declararea structurilor de date cu **typedef** și **struct**		
alocare dinamică cu **malloc** / **free**		
alocare dinamică cu **new** / **delete**		
alocarea dinamică pentru matrici se face cu 2 apeluri de malloc/new		
alocarea dinamică pentru matrici se face cu 1 apel de malloc/new		

Despre aplicație:

Proiectul este fără erori de compilare/linkeditare		
Programul a fost testat cu seturi de date acoperitoare		
Operațiile de intrare se fac cu verificare		
Programul rulează fără erori		
Programul rulează și întreaga memorie alocata este dealocata		

Problema II

Cerințe minimale:

Se preiau de la consolă doua șiruri de caractere: sir_1, sir_2 memorate în modul specific C pentru șiruri de caractere.

a. Să se calculeze lungimea șirurilor introduse;
b. Să se compare (în sens lexicografic) sir_1 cu sir_2 specificându-se prin 0 că cele două șiruri sunt egale, printr-o valoare pozitiva că sir1_1>sir_2 și printr-o valoare negativă că sir_1<sir_2 și tipărind un mesaj în acest sens;
c. Să se copie sir_1 într-un șir nou, sir_3;
d. Să se copie maxim n caractere din sir_1 în sir_3;
e. Să se concateneze sir_1 cu sir_2 și să se memoreze noul șir în sir_3;

Indicație: un șir de caractere se păstrează într-o zonă de memorie organizată ca tablou unidimensional de tip char. După ultimul caracter al șirului se păstrează caracterul NULL ('\0'). **NU** se vor folosi funcțiile din biblioteca standard ce permit operații cu șiruri de caractere!

Cerințe în completare (1):

Folosind pointeri la șiruri de caractere să se proiecteze :

a. o funcție care sa calculeze și să returneze lungimea șirului pentru care se apelează;
b. o funcție care să compare doua șiruri ce apar ca parametri, sir_1, sir_2, returnând 0 dacă cele două șiruri sunt egale, o valoare pozitiva dacă sir1_1>sir_2 și o valoare negativă dacă sir_1<sir_2;
c. o funcție care să copie un șir sursă într-un șir destinație, având ca prim parametru destinația și drept al doilea sursa;
d. o funcție care să copie cel mult n caractere dintr-un șir sursă într-un șir destinație, care să aibă ca parametri: șirul destinație, șirul sursă și numărul maxim de caractere;
e. o funcție care să concateneze două șiruri având ca prim parametru șirul sursă și drept al doilea, șirul care se adaugă la sfârșitul șirului sursă.

Indicație: NU se vor folosi funcțiile din biblioteca standard ce permit operații cu șiruri de caractere! Se cere implementarea unor funcții proprii similare celor cu prototipul descris în fișierul string.h: strlen, strcpy, strncpy, strcat, strcmp.

Cerințe în completare (2):

Se vor defini structuri de date pentru șiruri de caractere la care se memorează lungimea șirului. Se vor rescrie funcțiile pentru memorarea șirurilor de caractere.

Indicație: După ultimul caracter al șirului nu se mai păstrează caracterul NULL ('\0'). In structura de date se memorează câmpuri pentru lungimea șirului și pentru vectorul de caractere. Datele se pot memora în zone de memorie alocate dinamic.

Problema II (fișa de verificare)

Descriere implementare:

Soluția implementează:

Cerința	autoevaluare	evaluare
numai cerința minimală		
funcție pentru calculul lungimii șirului de caractere		
funcție pentru compararea a două șiruri de caractere		
funcție pentru copierea unui sir de caractere in altul		
funcție pentru copierea a maxim n caractere dintr-un șir de caractere în altul		
funcție pentru concatenarea a două șiruri de caractere		

Implementarea conține:

operații de intrare / ieșire cu **scanf** / **printf**		
operații de intrare / ieșire cu **gets** / **puts**		
operații de intrare / ieșire cu **cin** / **cout** și << / >>		
fișier header.h cu protecție la dublă includere		
declararea structurilor de date cu **typedef** și **struct** pentru un șir de caractere memorat cu lungimea șirului		
funcțiile sunt scrise pentru acest mod de reprezentare a șirurilor de caractere		
alocare dinamică cu **malloc** / **free**		
alocare dinamică cu **new** / **delete**		

Despre aplicație:

Proiectul este fără erori de compilare/linkeditare		
Aplicația conține exemple de apelare a funcțiilor implementate		
Aplicația nu utilizează funcțiile de bibliotecă		
Programul a fost testat cu seturi de date acoperitoare		
Programul rulează fără erori		
Programul rulează și întreaga memorie alocată este și dealocată		