Programare orientată obiect

Obiective

- **Cunoaşterea şi înţelegerea conceptelor specifice programării orientate obiect**
- **Abilități de programare în limbajele de programare C și C++**

Obiectivele specifice:

- Scrierea de programme de scară mică/mijlocie cu interfeţe grafice utilizator folosind C++ și QT.
- Proiectarea oriantată obiect pentru programe de scară mică/mijlocie
- Explicarea/Înțelegerea structurilor de tip clasă ca fiind componente fundamentale in construirea aplicatiilor.
- Înțelegerea rolului moștenirii, polimorfismului, legării dinamice și a structurilor generice în realizarea codului reutilizabil.
- Utilizarea claselor/modulelor scrise de alți programatori în dezvoltarea sistemelor proprii

Programare orientată obiect

Curs 1-2

- Limbajul C (sintaxă, instrucțiuni, tipuri de date,funcții)
- Programare modulară în C (funcții, module)

Curs 3 -5

- Clase și obiecte (c++, clase, atribure, metode, obiecte)
- Șabloane (template), Diagramă uml de clase
- Moștenire. Polimorfism

Curs 6-7

- Intrări/iesiri. Fișiere.
- Excepții. Tratarea excețiilor in c++

Curs 8-10

- Interfețe grafice utilizator (componente grafice, semnale/sloturi)
- Qt Designer, Componente grafice Qt cu model

Curs 11 -12

- Șabloane de proiectare,
- STL iteratori, STL Algorithm

Curs 13

- Șabloane de proiectare, Gestiunea memoriei, Smart pointer

Bibliografie

1. A.V. Aho, J.E. Hopcroft, J.D. Ullman, Data Structures and Algorithms, Addisson-Wessley Publ., Massachusetts, 1983.
2. R. Andonie, I. Garbacea, Algoritmi fundamentali. O perspectiva C++, Editura Libris,
3. Alexandrescu, Programarea moderna in C++. Programare generica si modele de proiectare aplicate, Editura Teora, 2002
4. M. Frentiu, B. Parv, Elaborarea programelor. Metode si tehnici moderne, Ed. Promedia, Cluj-Napoca, 1994.
5. E. Horowitz, S. Sahni, D. Mehta, Fundamentals of Data Structures in C++, Computer Science Press, Oxford, 1995.
6. K.A. Lambert, D.W. Nance, T.L. Naps, Introduction to Computer Science with C++, West Publishing Co., New-York, 1996.
7. L. Negrescu, Limbajul C++, Ed. Albastra,Cluj-Napoca 1996.

Cluj_Napoca, 1995.

8. Dan Roman, Ingineria programarii obiectuale, Editura Albastra, Cluj_Napoca, 1996.
9. B. Stroustup, The C++ Programming Language, Addison Wesley, 1998.
10. Bruce Eckel, Thinking in C++, www.bruceeckel.com

Note/Reguli

Calculul notei:

Notă Laborator 30%
Notă examen scris 30%
Notă examen practic 40%
Activitate seminar bonus de 0.5 pct

Pentru promovare este nevoie de minim 5 la fiecare notă.

Laborator/prezențe

Cei care nu au nota 5 la laborator sau nu satisfac criteriu cu numarul de prezență la laborator/seminar nu pot intra in examenul din sesiune.
Pot veni la examenul din sesiunea de restanță.
În restanță se predau laboratoare (cei care nu au luat minim 5 in timpul semestrului).
Nota maxima pentru nota pe activitatea de laborator este 5 in acest caz.

Restanță

În sesiunea de restanță se poate da examenul scris, examenul practic sau ambele. Valabil atât pentru cei care au picat examenul (examenele) cât și pentru cei care vin la mărire de notă.

Examenul scris

Examenul scris se da în timpul sesiunii.

Durata 1.5 – 2 ore

Reprezintă 30% din nota finală

Tipuri de probleme:

- sintaxă C++ , algoritmi
  - Se dă o funcție C++ - se cere specificare și testare pentru funcția dată
  - se dă specificația sau niște funcții de test - se cere implementarea c++

- concepte din programarea orientată obiect – C++ : alocare dinamică, constructor, destructor, moştenire, polimorfism, metode virtuale, clase abstracte, suprascriere, supraâncărcare, excepţii, streamul IO, containere, iteratori, algoritmi STL, etc.
  - Se dă un cod c++ în care se folosesc anumite concepte - se cere rezultatul rulării, identificarea erorilor
- UML – C++
  - se dă un cod c++ - se cere diagrama UML de clasă
  - Se dă o diagramă UML de clase – se cere codul c++
  - Se dă o descriere a unor clase și relațiile intre clase – se cere codul c++
- Qt
  - se da codul sursă Qt – se cere o schiţa a interfeţei grafice, explicații despre ce își propune codul dat
  - se da o schiță a interfeței grafice – se cere codul Qt care construieste interfața utilizator conform schiței

Examen practic

În aceeași zi cu examenul scris

Durată: 2.5 - 3 ore

Reprezintă 40% din nota finală

Se cere o aplicație cu interfață grafică utilizator (QT).

Aplicația se dezvoltă pornind de la un proiect gol, nu se pot folosi coduri surse externe (existente)

Se poate folosi:

- QT Assistant
- Pe o foaie A4 se pot scrie API - signaturi de metode, constante, operatori, include-uri,etc (fără algoritmi)

Aplicaţia de dezvoltat:

- Citește/scrie date din/in fișier text
- Validează datele introduse de utilizator
- Folosește arhitectura stratificată
- Specificații și teste
- 3-5 funcționalități
    - Se punctează doar acele funcționalități care se pot demonstra executând aplicaţia (nu se dau puncte pentru cod sursă)

Se pot folosi laptopuri proprii la examen - orice mediu de dezvoltare

Se pot folosi calculatoarele facultăţii - eclipse + Qt + Qt plugin

Sugestii pentru examen

Pentru pregătire - faceţi exerciţii în condiţii similare ca şi la simulare. Vedeţi cât timp vă ia, care sunt problemele de care vă loviţi.

Folosiţi dezvoltarea bazate pe teste. Construiţi aplicaţia incremental. Paşi mici, salvat-compilat-testat frecvent.

Adăugaţi căte o metodă odată să puteţi reveni uşor la o versiune anterioară care funcţiona.

Nu ignoraţi erorile, rezolvaţi problema înainte să treceţi mai departe. Nu treceţi mai departe dacă nu ştiţi dacă ultimul lucru adăugat funcţionează.

Nu ignorați warningurile, ele pot indica erori în program (ex. pointeri)

Folosiți containere , algoritmi STL.

Nu implementați lucruri care nu se cer, ele nu se vor puncta (Ex nu implementați ștergere dacă nu se cere în problemă). Nu se dau puncte pe volumul de cod sursă.

Dacă ceva nu va iese, încercați sa treceți peste, implementați o variantă banală (Ex. În loc să citească din fișier returnează niste obiecte hardcodate) și reveniți ulterior pentru a rezolva problema.

Dacă folosiţi laptopul propriu asiguraţivă ca funcţionează (încărcător, softurile necesare instalate şi funcţionale). Pentru examen pregătiţi un workspace gol care funcţionează corect.