**fişa disciplinei**

**1. Date despre program**

|  |  |
| --- | --- |
| 1.1 Instituţia de învăţământ superior | **Universitatea Babeş-Bolyai Cluj-Napoca** |
| 1.2 Facultatea | **Facultatea de Matematică şi Informatică** |
| 1.3 Departamentul | **Departamentul de informatică** |
| 1.4 Domeniul de studii | **Informatică** |
| 1.5 Ciclul de studii | **Licenţă** |
| 1.6 Programul de studiu / Calificarea | **Informatică (în limba germană)** |

**2. Date despre disciplină**

|  |  |
| --- | --- |
| 2.1 Denumirea disciplinei | **Programare orientată obiect** |
| 2.2 Titularul activităţilor de curs | Ioan Crișan |
| 2.3 Titularul activităţilor de seminar | Ioan Crișan |
| 2.4 Anul de studiu | **1** | 2.5 Semestrul | **2** | 2.6. Tipul de evaluare | **E** | 2.7 Regimul disciplinei | **Obligatorie** |
| 2.8. Codul disciplinei | **MLG5006** |

**3. Timpul total estimat** (ore pe semestru al activităţilor didactice)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 3.1 Număr de ore pe săptămână | 5 | Din care: 3.2 curs | 2 | 3.3 seminar/laborator | 1sem+2 lab |
| 3.4 Total ore din planul de învăţământ | 70 | Din care: 3.5 curs | 28 | 3.6 seminar/laborator | 14+28 |
| Distribuţia fondului de timp: | ore |
| Studiul după manual, suport de curs, bibliografie şi notiţe | 20 |
| Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate şi pe teren | 20 |
| Pregătire seminarii/laboratoare, teme, referate, portofolii şi eseuri | 20 |
| Tutoriat | 14 |
| Examinări  | 6 |
| Alte activităţi: .................. | - |
| 3.7 Total ore studiu individual | 80 |
| 3.8 Total ore pe semestru | 150 |
| 3.9 Numărul de credite | 6 |

**4. Precondiţii** (acolo unde este cazul)

|  |  |  |
| --- | --- | --- |
| 4.1 de curriculum | Fundamentele Programării, Structuri de date și algoritmi |  |
| 4.2 de competenţe | Cunoștințe medii de programare ȋntr-un limbaj de programare de nivel ȋnalt |  |

**5. Condiţii** (acolo unde este cazul)

|  |  |
| --- | --- |
| 5.1 De desfăşurare a cursului | * Sală de curs cu videoproiector
 |
| 5.2 De desfăşurare a seminarului/laboratorului | * Sală de laborator cu calculatoare dotate cu limbajul de programare C++ și mediul de programare QT
 |

**6. Competenţele specifice acumulate**

|  |  |
| --- | --- |
| **Competenţe profesionale** | C1.1 Descrierea adecvată a paradigmelor de programare şi a mecanismelor de limbaj specifice, precum şi identificarea diferenţei dintre aspectele de ordin semantic şi sintactic.C1.2 Explicarea unor aplicaţii soft existente, pe niveluri de abstractizare (arhitectură, pachete, clase, metode) utilizând in mod adecvat cunoştinţele de bază C1.3 Elaborarea codurilor sursă adecvate şi testarea unitară a unor componente într-un limbaj de programare cunoscut, pe baza unor specificaţii de proiectare dateC1.4 Testarea unor aplicații pe baza unor planuri de testC1.5 Dezvoltarea de unități de program și elaborarea documentațiilor aferente |
| **Competenţe transversale** | CT1 Aplicarea regulilor de muncă organizată şi eficientă, a unor atitudini responsabile faţă de domeniul didactic-ştiinţific, pentru valorificarea creativă a propriului potenţial, cu respectarea principiilor şi a normelor de etică profesionalăCT3 Utilizarea unor metode şi tehnici eficiente de învăţare, informare, cercetare şi dezvoltare a capacităţilor de valorificare a cunoştinţelor, de adaptare la cerinţele unei societăţi dinamice și de comunicare în limba română și într-o limbă de circulație internațională |

**7. Obiectivele disciplinei** (reieşind din grila competenţelor acumulate)

|  |  |
| --- | --- |
| 7.1 Obiectivul general al disciplinei | * Să deprindă studentul cu proiectare orientată obiect a problemelor de scară mică/mijlocie și ȋnvățarea limbajului de programare C++ și a bibliotecii QT.
 |
| 7.2 Obiectivele specifice | * Demonstrarea diferenței dintre proiectarea imperativă tradițională și proiectarea orientată obiect.
* Explicarea structurilor de tip clasă ca fiind componente fundamentala, modulare.
* Înțelegerea rolului moștenirii, polimorfismului, legării dinamice și a structurilor generice ȋn realizarea codului reutilizabil.
* Exlicarea și utilizarea strategiilor de programare DEFENSIVE, utilizarea aserțiunilor formale și tratarea excepțiilor.
* Scrierea de programe de scară mică/mijlocie folosind C++ și QT.
* Utilizarea claselor scrise de alți programatori ȋn dezvoltarea sistemelor.
 |

**8. Conţinuturi**

|  |  |  |
| --- | --- | --- |
| **8.1 Curs** | Metode de predare | Observaţii |
| 1. **Paradigma de programare orientată obiect.**
* Elemente de bază ale limbajului C..
* Elemente lexicale. Operatori. Conversii.
* Tipuri de date. Variabile. Constante.
* Domeniul de vizibilitate și durata de viață a variabilelor. Spații de nume.
* Declararea și definirea funcțiilor. Supraȋncărcarea funcțiilor. Funcții inline.
 | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Exemple
* Demonstraţia didactică
 |  |
| **2. Programare modulară ȋn C++.*** Funcții. Parametrii.
* Fișiere header. Biblioteci.
* Implementarea modulară a TAD-urilor..
* Utilizarea pointerului void pentru obținerea genericității.
 | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Exemple
* Demonstraţia didactică
 |  |
| **3. Tipuri de date derivate și tipuri definite de utilizator, alocare dinamică ȋn C++.** * Tipuri de date: vectori și structuri.
* Tipuri de date: pointeri și referințe.
* Alocarea și dealocarea memoriei.
* Pointeri la funcții și pointeri spre void.
 | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
 |  |
| **4. Programare orientată obiect ȋn C++.** * Clase și obiecte.
* Membri unei clase. Modificatori de acces.
* Constructori/destructori.
* Diagrame UML pentru clase (membri, acces).
 | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
 |  |
| **5. Programare generică în C++**  | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
 |  |
| **6. Gestiunea resurselor și a memoriei în C++** | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
 |  |
| **7. Moștenire** * Moștenire simplă. Clase derivate.
* Principiul substituției.
* Supraȋncărcarea metodelor.
* Moștenire multiplă.
* Relații de specializare/generalizare – reprezentări UML.
 | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
 |  |
| **8. Polimorfism** | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
 |  |
| **9. Interfață utilizator.** | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
 |  |
| **10. Elemente de programare bazată pe evenimente I.** | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
 |  |
| **11. Elemente de programare bazată pe evenimente II.** | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
 |  |
| **12. Biblioteca STL*** Iteratori STL.
* Algoritmi STL.
 | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
 |  |
| **13. Aplicație POS (Point Of Sale)** * Șabloane de proiectare Façade, Strategy.
* Șablonul de proectare Composite.
 | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
 |  |
| **14. Recapitulare** | * Expunerea interactivă
* Conversaţia
 |  |
| **Bibliografie**1. Bruce Eckel, Thinking in C++, www.bruceeckel.com
2. Alexandrescu, Programarea moderna in C++. Programare generica si modele de proiectare aplicate, Editura Teora, 2002
3. M. Frentiu, B. Parv, Elaborarea programelor. Metode si tehnici moderne, Ed. Promedia, Cluj-Napoca, 1994.

**In limba germana:**1. G. Goos, W. Zimmermann, Objektorientiertes Programmieren und Algorithmen, Springer, Berlin, Heidelberg, New York, 2006.
2. Pötzsch-Heffter, A., Konzepte objektorientierter Programmierung, Springer, Berlin, Heidelberg, 2009.
3. Küchlin, W, Weber, A.,Einführung in die Informatik, Objektorientiertes Programmieren mit Java, Springer, Berlin, Heidelberg, New York, 2004.
4. B. Stroustup, Die C++ Programmiersprache, Addison Wesley, 2000.
 |
| **8.2 Seminar** | Metode de predare | Observaţii |
|  |  | Seminarul este structurat sub forma a 2 ore din 2 în 2 săptămâni. |
| 1. Probleme simple ȋn C. Funcții și parametric. Variabile locale și globale. Vizibilitate. Vectori (uni și multidimensionali) și structuri. | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
* Exemple
 |  |
| 2. TAD de tip container cu elemente generice (void\*): reprezentări vizibile și ascunse. | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
* Exemple
 |  |
| 3. Clase. Clase simple. Supraȋncărcarea operatorilor. Clase cu obiecte ca date membre. | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
* Exemple
 |  |
| 4. Clase de tip vector dinamic și iteratori. Moștenire. | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
* Exemple
 |  |
| 5. Clase abstracte și interfețe. Polimorfism. | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
* Exemple
 |  |
| 6. Clase: template și excepții | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
* Exemple
 |  |
| 7. Probleme complexe implementate pe baza diagramelor UML. Șabloane de proiectare. Pregătire pentru examenul scris. | * Expunerea interactivă
* Explicaţia
* Conversaţia
* Demonstraţia didactică
* Exemple
 |  |
| **8.3 Laborator** | Metode de predare | Observaţii |
|  |  | * Laboratorul este structurat sub forma a 2 ore săptămânal.
* Documentațiile de laborator se predau ȋn săptămȃna următoare celei ȋn care s-a dat tema, iar programele se predau după două săptămȃni.
 |
| 1. Instalare Visual Studio. Specificare, proiectare și implementare probleme simple ȋn C/C++. Aspecte generale C/C++.  | * Lucrare de laborator
* Explicaţia
* Conversaţia
* Modelarea
 |  |
| 2. Programare modulară ȋn C++ | * Lucrare de laborator
* Explicaţia
* Conversaţia
* Modelarea
 |  |
| 3. Proces de dezvoltare “feature driven”  | * Lucrare de laborator
* Explicaţia
* Conversaţia
* Modelarea
 |  |
| 4. Proces de dezvoltare “feature driven”  | * Lucrare de laborator
* Explicaţia
* Conversaţia
* Modelarea
 |  |
| 5. Proces de dezvoltare “feature driven”  | * Lucrare de laborator
* Explicaţia
* Conversaţia
* Modelarea
 |  |
| 6. Arhitecturi stratificate | * Lucrare de laborator
* Explicaţia
* Conversaţia
* Modelarea
 |  |
| 7. Arhitecturi stratificate | * Lucrare de laborator
* Explicaţia
* Conversaţia
* Modelarea
 |  |
| 8. Arhitecturi stratificate | * Lucrare de laborator
* Explicaţia
* Conversaţia
* Modelarea
 |  |
| 9. Fișiere text | * Lucrare de laborator
* Explicaţia
* Conversaţia
* Modelarea
 |  |
| 10. GUI folosind QT | * Lucrare de laborator
* Explicaţia
* Conversaţia
* Modelarea
 |  |
| 11. Repository. | * Lucrare de laborator
* Explicaţia
* Conversaţia
* Modelarea
 |  |
| 12. Containere, iteratori și algoritmi STL | * Lucrare de laborator
* Explicaţia
* Conversaţia
* Modelarea
 |  |
| 13. Predare laboratoare (a se vedea observația) | * Lucrare de laborator
* Explicaţia
* Conversaţia
* Modelarea
 |  |
| 14. Predare laboratoare (a se vedea observația) | * Lucrare de laborator
* Explicaţia
* Conversaţia
* Modelarea
 |  |
| **Bibliografie**1. Bruce Eckel, Thinking in C++, www.bruceeckel.com
2. Alexandrescu, Programarea moderna in C++. Programare generica si modele de proiectare aplicate, Editura Teora, 2002
3. M. Frentiu, B. Parv, Elaborarea programelor. Metode si tehnici moderne, Ed. Promedia, Cluj-Napoca, 1994.

**In limba germana:**1. G. Goos, W. Zimmermann, Objektorientiertes Programmieren und Algorithmen, Springer, Berlin, Heidelberg, New York, 2006.
2. Pötzsch-Heffter, A., Konzepte objektorientierter Programmierung, Springer, Berlin, Heidelberg, 2009.
3. Küchlin, W, Weber, A.,Einführung in die Informatik, Objektorientiertes Programmieren mit Java, Springer, Berlin, Heidelberg, New York, 2004.
4. B. Stroustup, Die C++ Programmiersprache, Addison Wesley, 2000.
 |

**9. Coroborarea conţinuturilor disciplinei cu aşteptările reprezentanţilor comunităţii epistemice, asociaţiilor profesionale şi angajatori reprezentativi din domeniul aferent programului**

|  |
| --- |
| * Conţinutul disciplinei este în concordanţă cu recomandările ACM pentru domeniul informatică.
* Cursul există ȋn programul des tudiu al universităților importante din țară și străinătate.
* Conţinutul disciplinei este considerat de companiile software ca important pentru a asigura cunoștințe medii de programare orientată obiect.
 |

**10. Evaluare**

|  |  |  |  |
| --- | --- | --- | --- |
| Tip activitate | 10.1 Criterii de evaluare | 10.2 metode de evaluare | 10.3 Pondere din nota finală |
| 10.4 Curs | * Corectitudinea și completitudinea cunoștințelor acumulate și capacitatea de a proiecta și implementa programe C++
 | Examen scris (ȋn sesiunea regulară) | 40% |
| 10.5 Seminar/Laborator | * Capacitatea de a proiecta, testa și depana programe C++ folosind QT
 | Evaluare practică (ȋn sesiunea regulară) | 30% |
| * Corectitudinea programelor C++ și a documentațiilor de laborator
 | -documentații-portofoliu-observare continuă | 30% |
| 10.6 Standard minim de performanţă |
| * Fiecare student trebuie să demonstreze că a atins un nivel acceptabil de cunoaştere şi înţelegere a domeniului, că este capabil să exprime cunoştinţele într-o formă coerentă, că are capacitatea de a stabili anumite conexiuni şi de a utiliza cunoştinţele în rezolvarea unor probleme ȋn limbajul de programare C++.
* Pentru promovare este necesar ca nota finală să fie minim 5.
 |

Data completării Semnătura titularului de curs Semnătura titularului de seminar

14.04.2024 Ioan Crișan Ioan Crișan

Data avizării în departament Semnătura directorului de departament

 Conf. dr. Adrian Sterca