Academia de Studii Economice
Departamentul de Informatică şi Cibernetică Economică
Calea Dorobanţi, 15-17, Sector 1, Bucureşti, 010552 (camera 2314)
Tel.: +40 21 319 19 00, ext. 319, 336, Fax: +40 21 311 20 66
www.dice.ase.ro

Tematica de concurs pentru postul de Asistent universitar (perioadă determinată 1 an), poziția 147, 2020-2021, semestrul 2

Discipline: Programare Orientată Obiect, Structuri de date, Calitate și testare software

Programare Orientată Obiect

1. Elemente ale programării procedurale: funcții, transferul parametrilor, pointeri la date si funcții, clase de memorie.
2. Conceptele de clasa, obiect, constructor, destructor, metode de acces, pointerul *this*. Declararea si implementarea metodelor in clasa si in afara clasei.
3. Asimilarea conceptelor de constructor de copiere, supraîncărcare operator =, obiecte cu extensii in memoria dinamica si domenii de nume (namespace).
4. Conversii intre diferite tipuri de obiecte (operatorul cast, operatorul= si constructor de copiere), vector de obiecte, modificatorul const, tipologia membrilor statici (static), obiecte constante, pointeri constanți la obiecte si pointeri la obiecte constante.
5. Mecanismul try-catch in C++.
6. Supraîncărcarea operatorilor.
7. Clase derivate, moștenire. Polimorfism.
8. Funcții virtuale, supradefinire, mostenire multipla.
9. Mecanisme de tip RTTI, mostenire multipla si dynamic cast.
10. Funcții si clase template.
11. Operații I/O orientate pe stream-uri. Conceptele de serializare/deserializare obiecte.
12. Standard Template Library - STL – containere, iteratori si algoritmi. Clasa string, map, list, vector, etc.

Bibliografie:

1. Ion Smeureanu, Marian Dardala, Programarea orientata obiect in limbajul C++, CISON, Bucuresti, 2002, România
2. Ion Smeureanu, Marian Dardala, Programarea in limbajul C/C++, CISON, Bucuresti, 2001, România
3. Herbert Schildt, C++ manual complet, Teora, Bucuresti, România
4. Bjarne Stroustroup , The C++ Programming Language, 3rd Edition, Addison-Wesley

Structuri de date

1. Tipuri de date standard si definite de programator. Pointeri. Modele si cerințe de definire, inițializare, utilizare si lizibilitate a datelor in programul sursa. Indicatori de performanta ai utilizării memoriei. Memoria STACK si memoria HEAP.

2. Structuri de date dinamice necontigue: lista simplu înlănțuită și lista dublu înlănțuită – definire, alocare și utilizare. Stiva și coada – definire, alocare și utilizare.
3. Matrice rare: definire, alocare și utilizare. Structuri de date neomogene și contigue. Implementarea matricelor rare prin masive și structuri de date neomogene.
4. Structura de date de tip graf: caracteristici, definire, alocare și utilizare. Algoritmi de traversare a unui graf. Conectivitate și algoritmi de conectivitate.
5. Structuri de date arborescente: arbore oarecare și arbore binar – definire, alocare și utilizare. Arbori de structură: caracteristici, implementare și operații.
6. Structuri de date arborescente: arbore binar de căutare – definire, alocare și utilizare.
7. Structuri arborescente echilibrate: arbori binari echilibrați, arbori AVL, arbori Roşu şi Negru – definire operație, caracteristici ale arborilor echilibrați.
8. Structura de date Arbore B: definire, proprietăți, alocare, algoritmi şi implementarea operațiilor de bază (inserare și ștergere).
9. Tabele de dispersie: caracteristici, funcții hash, operații, mecanisme de evitare a coliziunilor.
10. Structura Heap – definire, alocare și utilizare. Cozi de prioritate.
11. Compactarea și compresia datelor: caracteristici, clasificare şi algoritmi de compresie a datelor. Conversii ale structurilor de date: caracteristici ale procesului de conversie, tipuri de conversie, elemente de eficiență la nivel de proces, respectiv la nivel de aplicație.

Bibliografie:

1. Ion Ivan, Marius Popa, Paul Pocatilu (coordonatori) , *Structuri de date*, Editura ASE, București, 2008, România
2. Ion Smeureanu, Marian Dardala, Programarea in limbajul C/C++, CISON, Bucureşti, 2001, România
3. Saumeyendra Sengupta, Carl Phillip Korobkin , *C++ Object Oriented Data Structures*, Springer Verlag, New York, 1994, Statele Unite ale Americii
4. William Ford, William Topp, Data Structures with C++, Prentice Hall , New Jersey, 1996, Statele Unite ale Americii
5. Demaine, *Advanced Data Structures*, 2003, http://courses.csail.mit.edu/6.897/spring03/scribe_notes, Statele Unite ale Americii

Calitate și Testare Software

1. Principii privind scrierea codului sursă - Clean Code (SOLID, DRY, KISS, YAGNI, WET, convenții de nume, definirea și implementarea funcțiilor)
2. Design Patterns: Singleton, Simple Factory, Factory Method, Abstract Factory, Builder, Adapter, Decorator, Facade, Flyweight, Chain of Responsability, Command, Observer, State, Strategy, Memento
3. Gestiunea versiunilor codului sursa - Git, SVN
4. Concepte privind testarea unitară (Unit Testing)

Academia de Studii Economice
Departamentul de Informatică şi Cibernetică Economică
Calea Dorobanţi, 15-17, Sector 1, Bucureşti, 010552 (camera 2314)
Tel.: +40 21 319 19 00, ext. 319, 336, Fax: +40 21 311 20 66
www.dice.ase.ro

5. Utilizarea framework-ului JUnit (versiunile 3, 4 si 5)
6. Concepte privind calitatea software - metrici, indicatori, instrumente, testare automata (testare aplicații Web cu platforma Selenium)

Bibliografie:

1. Documentatie JUnit disponibila la adresa http://junit.org
2. Scott Chacon, Bean Straub - Pro Git, 2nd edition, Apress, 2014, disponibila online la adresa http://git-scm.com/book/en/v2
3. Robert C. Martin - *Clean Code, A Handbook of Agile Software Craftsmanship*, Prentice Hall, 2009
4. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides - *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1995
5. Steve Holzner, *Design Patterns for Dummies*, Wiley, 2006
6. Lasse Koskela, *Effective Unit Testing*, Manning, 2013
7. Lasse Koskela, *Practical TDD and Acceptance TDD for Java Developers*, Manning, 2007
8. Andrew Hunt, David Thomas, *Pragmatic Unit Testing in Java with JUnit*, The Pragmatic Programmers, 2003

Prof. univ. dr. Ion SMEUREANU