

# **PROGRAMA**

DE EXAMEN PENTRU DISCIPLINA: **INFORMATICĂ**

**Specializarea matematică-informatică**

**BACALAUREAT 2010**

**Aprobată prin O.M.E.C.I. nr. \_5508/06.10.2009\_**

**2009-2010**

## **I. STATUTUL DISCIPLINEI**

Disciplina INFORMATICĂ are statutul de disciplină opțională la proba "E".

## **II. COMPETENȚE DE EVALUAT**

- construirea algoritmilor corespunzători unor prelucrări elementare și reprezentarea lor prin intermediul programelor pseudocod și programelor scrise în limbaj de programare (Pascal sau C/C++, la alegere);
- analiza rezolvării unei probleme prin urmărirea evoluției valorilor variabilelor prelucrate de algoritmul corespunzător;
- abstractizarea rezolvării prin construirea unor algoritmi echivalenți;
- identificarea și utilizarea tipurilor de date predefinite specifice unui limbaj de programare;
- definirea și utilizarea unor tipuri de date proprii;
- identificarea și utilizarea operatorilor predefiniți elementari;
- identificarea și utilizarea subprogramelor predefinite elementare;
- identificarea și utilizarea regulilor sintactice specifice limbajului de programare studiat;
- definirea și apelul unor subprograme proprii cu înțelegerea mecanismelor de transfer prin intermediul parametrilor;
- identificarea proprietăților unor structuri de date necesare în rezolvarea problemelor cu ajutorul calculatorului și utilizarea unor modele de memorare a acestora;
- organizarea datelor ce intervin în rezolvarea unei probleme utilizând structuri de date adecvate;
- organizarea etapelor de prelucrare ce formează un algoritm utilizând structuri de control și module de program;
- folosirea unor metode sistematice de rezolvare pentru probleme de generare;
- analiza unor algoritmi echivalenți de rezolvare a unei probleme în vederea alegerii algoritmului optim.

## **III. CONȚINUTURI**

### **1. Algoritmi**

- 1.1. Noțiunea de algoritm, caracteristici
  - 1.2. Date, variabile, expresii, operații
  - 1.3. Structuri de bază (liniară, alternativă și repetitivă)
  - 1.4. Descrierea algoritmilor (programe pseudocod)
- 2. Elementele de bază ale unui limbaj de programare (Pascal sau C, la alegere)**
- 2.1. Vocabularul limbajului
  - 2.2. Constante. Identificatori
  - 2.3. Noțiunea de tip de dată. Operatori aritmetici, logici, relaționali
  - 2.4. Definirea tipurilor de date
  - 2.5. Variabile. Declararea variabilelor
  - 2.6. Definirea constantelor
  - 2.7. Structura programelor. Comentarii
  - 2.8. Expresii. Instrucțiunea de atribuire
  - 2.9. Citirea/scrierea datelor
  - 2.10. Structuri de control (instrucțiunea compusă, structuri alternative și repetitive)
- 3. Subprograme predefinite**
- 3.1. Subprograme. Mecanisme de transfer prin intermediul parametrilor
  - 3.2. Proceduri și funcții predefinite
- 4. Tipuri structurate de date**
- 4.1. Tipul tablou
  - 4.2. Tipul șir de caractere
    - operatori, proceduri și funcții predefinite pentru: citire, afișare, concatenare, căutare, extragere, inserare, eliminare și conversii (șir ↔ valoare numerică)
  - 4.3. Tipul înregistrare
- 5. Fișiere text**
- 5.1. Fișiere text. Tipuri de acces
  - 5.2. Proceduri și funcții predefinite pentru fișiere text
- 6. Algoritmi elementari**
- 6.1. Probleme care operează asupra cifrelor unui număr
  - 6.2. Divizibilitate. Numere prime. Algoritmii lui Euclid
  - 6.3. Șirul lui Fibonacci. Calculul unor sume cu termenul general dat
  - 6.4. Determinare minim/maxim
  - 6.5. Metode de ordonare (metoda bulelor, inserției, selecției, numărării)
  - 6.6. Interclasare
  - 6.7. Metode de căutare (secvențială, binară)
  - 6.8. Analiza complexității unui algoritm (considerând criteriile de eficiență *durata de executare și spațiu de memorie utilizat*)
- 7. Subprograme definite de utilizator**
- 7.1. Proceduri și funcții
    - declarare și apel
    - parametri formali și parametri efectivi

- parametri transmiși prin valoare, parametri transmiși prin referință
- variabile globale și variabile locale, domeniu de vizibilitate

7.2. Proiectarea modulară a rezolvării unei probleme

## 8. Recursivitate

- 8.1. Prezentare generală
- 8.2. Proceduri și funcții recursive

## 9. Metoda backtracking (iterativă sau recursivă)

- 9.1. Prezentare generală
- 9.2. Probleme de generare

## 10. Generarea elementelor combinatoriale

- 10.1. Permutări, aranjamente, combinări
- 10.2. Produs cartezian, submulțimi

## 11. Liste

- 11.1. Structuri de date înlănțuite (reprezentări grafice, reprezentări statice în memorie)
- 11.2. Liste liniare simplu înlănțuite (definire și operații: inserare, căutare, eliminare)
- 11.3. Liste particulare (stive, cozi, liste circulare) și operații specifice

## 12. Grafuri

- 12.1. Grafuri neorientate
  - terminologie (nod/vârf, muchie, adiacență, incidență, grad, lanț, lanț elementar, ciclu, ciclu elementar, lungime, subgraf, graf parțial)
  - proprietăți (conexitate, componentă conexă, graf complet)
  - metode de reprezentare (matrice de adiacență, liste de adiacență)
- 12.2. Grafuri orientate
  - terminologie (nod/vârf, arc, adiacență, incidență, grad intern și extern, drum, drum elementar, circuit, circuit elementar, lungime, subgraf, graf parțial)
  - proprietăți (tare conexitate, componentă tare conexă)
  - metode de reprezentare (matrice de adiacență, liste de adiacență)
- 12.3. Arbori
  - terminologie (nod, muchie, rădăcină, descendent, descendent direct/fiu, ascendent, ascendent direct/părinte, frați, nod terminal, frunză)
  - metode de reprezentare în memorie (matrice de adiacență, liste "de descendenți", vector "de tați")