Examenul de bacalaureat național 2020
Proba E. d)
INFORMATICĂ
Limbajul C/C++

MODEL

Filieră teoretică, profil real, specializare matematică-informatică / matematică-informatică intensiv informatică
Filieră vocațională, profil militar, specializare matematică-informatică

- **Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.**
- **Timpul de lucru efectiv este de 3 ore.**
- **Identificatorii utilizați în rezolvări trebuie să respecte precizările din enunț (bold), iar în lipsa unor precizări explicite, notațiile trebuie să corespundă cu semnificațiile asociate acestora (eventual în formă prescurtată). Datele de intrare se consideră corecte, validarea lor nefiind necesară.**
- **În grafurile din cerințe oricare arc/muchie are extremități distincte și oricare două arce/muchii diferă prin cel puțin una dintre extremități.**

SUBIECTUL I **(20 de puncte)**

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Variabilele **x**, **y** și **z** sunt de tip întreg și memorează numere naturale din intervalul **[1,10³]**. Indicați o expresie C/C++ care are valoarea **1** dacă și numai dacă valoarea variabilei **x** este strict mai mică decât valoarea oricăreia dintre variabilele **y** și **z**.

a. `z+x<x+y && x+z>z+y` **b.** `z+x<x+y && z+y>y+x`

c. `x+z<z+y && z+y>y+x` **d.** `x+y<y+z && x+z>z+y`

2. Subprogramele **f1**, **f2** și **f3** sunt definite mai jos.

```
int f1(int n)
{ if(n==0) return 1;
  return n*f1(n-1);
}
```

```
int f2(int n)
{  if(n>1)
     return n*(n-1)*f2(n-2);
   return 1;
}
```

```
int f3(int n)
{ int f=1;
  while(n!=0)
  { f=f*n;
    n=n-1;
  }
  return f;
}
```

Pentru **n=12**, se obține aceeași valoare la apelul subprogramelor:

a. **f1** și **f2** **b.** **f1** și **f3** **c.** **f2** și **f3** **d.** **f1**, **f2** și **f3**

3. Având la dispoziție cinci tipuri de prăjituri, cu **caise**, cu **căpșune**, cu **prune**, cu **piersici**, respectiv cu **cireșe**, se utilizează metoda backtracking pentru a obține toate posibilitățile de a forma platouri cu câte trei tipuri de prăjituri diferite, știind că în cadrul unui platou nu contează ordinea de așezare a prăjiturilor și că prăjiturile cu **căpșune** nu vor fi plasate pe același platou cu prăjiturile cu **piersici**. Primele patru soluții obținute sunt, în această ordine: (**caise**, **căpșune**, **prune**), (**caise**, **căpșune**, **cireșe**), (**caise**, **prune**, **piersici**), (**caise**, **prune**, **cireșe**). A șasea soluție generate este:

a. {**caise**, **prune**, **căpșune**} **b.** {**caise**, **piersici**, **cireșe** }

c. {**căpșune**, **prune**, **cireșe**} **d.** {**prune**, **piersici**, **cireșe**}

4. Un arbore cu **8** noduri, numerotate de la **1** la **8**, are drept rădăcină nodul numerotat cu **5** și muchiile **[1,5]**, **[2,7]**, **[3,7]**, **[3,6]**, **[4,5]**, **[5,7]**, **[7,8]**. Indicați numărul de noduri care sunt descendenți direcți („fii") ai nodului **7**.

a. **2** **b.** **3** **c.** **4** **d.** **5**

5. Un graf orientat are **10** arce, **3** componente tare conexe, iar fiecare vârf al său are atât gradul interior, cât și gradul exterior nenule. Numărul minim de noduri pe care le poate avea graful este:

a. **4** **b.** **5** **c.** **6** **d.** **7**

SUBIECTUL al II-lea **(40 de puncte)**

1. Algoritmul alăturat este reprezentat în pseudocod.
S-a notat cu **a%b** restul împărțirii numărului natural **a** la numărul natural nenul **b**.

```
citește m,n,x
  (numere naturale nenule, m≤n)
 s←0; pm←1; pn←1
┌repetă
│┌dacă m%x=0 atunci
│| s←s+m; pm←x
│└■
│┌dacă n%x=0 și m≠n atunci
│| s←s+n; pn←x
│└■
│ m←m+pm
│ n←n-pn
└până când m>n
 scrie s
```

a. Scrieți valoarea afișată în urma executării algoritmului dacă se citesc, în această ordine, numerele **21**, **38** și **4**. **(6p.)**

b. Dacă pentru **m** și **x** se citesc numerele **20**, respectiv **2020**, scrieți cea mai mică și cea mai mare valoare care pot fi citite pentru variabila **n**, astfel încât, pentru fiecare dintre acestea, în urma executării algoritmului, să se afișeze **2020**. **(6p.)**

c. Scrieți programul C/C++ corespunzător algoritmului dat. **(10p.)**

d. Scrieți în pseudocod un algoritm, echivalent cu cel dat, înlocuind structura **repetă...până când** cu o structură de alt tip. **(6p.)**

2. Variabila **p** memorează, pentru fiecare dintre cele **20** de zone de parcare ale unui oraș, următoarele date specifice: identificatorul zonei, numărul de locuri închiriate pe parcursul lunii curente, precum și prețul practicat în zona respectivă pentru închirierea unui loc pentru o lună. Știind că expresiile C/C++ de mai jos au ca valori câte un număr natural, reprezentând identificatorul primei zone, respectiv suma obținută în urma închirierii pe parcursul lunii curente a tuturor locurilor de parcare din această zonă, scrieți definiția unei structuri cu eticheta **parcare**, care să permită memorarea datelor specifice unei zone de parcare, și declarați corespunzător variabila **p**.

p[0].id **p[0].numar*p[0].pret** **(6p.)**

3. Variabila **i** este de tip întreg, iar variabila **a** memorează un tablou bidimensional cu **5** linii și **7** coloane, numerotate începând cu **0**, cu elemente numere întregi. Fără a utiliza alte variabile decât cele menționate, scrieți o secvență de instrucțiuni în urma executării căreia să se afișeze pe ecran, separate prin câte un spațiu, indicii liniilor cu proprietatea că primul sau ultimul lor element are valoarea **2020**. **(6p.)**

SUBIECTUL al III-lea **(30 de puncte)**

1. Subprogramul **duplicare** are doi parametri:
- **n**, prin care primește un număr natural (**$n \in [1, 10^4]$**);
- **d**, prin care furnizează numărul obținut prin duplicarea fiecărei cifre impare a lui **n** sau **−1** dacă acesta nu are nicio cifră impară.

Scrieți definiția completă a subprogramului.
Exemplu: dacă **n=2019**, după apel **d=201199**. **(10p.)**

2. Un text are cel mult **100** de caractere, iar cuvintele sale sunt formate numai din litere mici ale alfabetului englez și sunt separate prin câte un spațiu. Scrieți un program C/C++ care citește de la tastatură un număr natural **n** (**$n \in [1, 10^2]$**), apoi un text de tipul precizat mai sus, și afișează pe ecran cuvintele acestuia, pe rânduri separate, astfel încât primele poziții să fie ocupate de mulțimea formată de cele care au cel puțin **n** litere, iar următoarele poziții, în continuarea acestora, să fie ocupate de mulțimea celorlalte cuvinte.

Cuvintele din aceeași mulțime sunt afișate într-o ordine oarecare, iar dacă una dintre cele două mulțimi este vidă, se afișează pe ecran doar mesajul **nu exista**.
Exemplu: pentru **n=5** și textul **el mergea tot spre aleea pietruita** datele afișate pot fi cele alăturate. **(10p.)**

```
mergea
aleea
pietruita
el
tot
spre
```

3. Fișierul **numere.in** conține pe prima linie un număr natural **n** (**$n \in [2, 10^9]$**), iar pe a doua linie un șir de cel mult **10^9** numere naturale din intervalul **[1,n]**. Numerele din șir sunt ordonate descrescător și sunt separate prin câte un spațiu. Se cere să se determine numărul valorilor naturale distincte din intervalul **[1,n]** care **NU** se găsesc în șirul menționat mai sus. Numărul determinat se afișează pe ecran. Proiectați un algoritm eficient din punctul de vedere al spațiului de memorie și al timpului de executare.
Exemplu: dacă fișierul conține numerele
10
8 8 8 5 3 3
se afișează pe ecran **7** (în șir nu se găsesc valorile **10 9 7 6 4 2 1**).

a. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. **(2p.)**

b. Scrieți programul C/C++ corespunzător algoritmului proiectat. **(8p.)**