BAREM – VARIANTA I

OFICIU ..... 10 puncte

SUBIECTUL I ..... 35 puncte

1. Rază ..... 20 puncte

Varianta 1: determinarea corectă a valorii ***nrSchimb*** bazată pe utilizarea *cmmdc*(***a***, ***b***) ..... 20 puncte

- *cmmdc*(***a***, ***b***) (sau *cmmmc*(***a***,***b***)) ..... 5 puncte
- calculul valorii ***nrSchimb*** ..... 15 puncte

Varianta 2: determinarea corectă a valorii ***nrSchimb*** cu alt algoritm corect (simulare) ..... 15 puncte

2. Numere cu „forță” ..... 15 puncte

- stabilirea forței unui număr ..... 5 puncte
- determinarea numărului grupurilor de elemente cu aceeași forță (***nrGr***) și a componenței lor (***grupuri***) ..... 10 puncte

SUBIECTUL II ..... 15 puncte

Numărul numerelor *Fibonacci* mai mici decât numărul *nr* dat

Cerința a.

- enunț problemă ..... 5 puncte

Cerința b.

- rezultat calculat corect (7) ..... 3 puncte

Cerința c.

- algoritm ..... 3 puncte
- autoapel corect ..... 2 puncte
- condiție de oprire din recursivitate ..... 2 puncte

SUBIECTUL III ..... 40 puncte

Prefix

Subprograme:

- citirea datelor de intrare ..... 3 puncte
- determinarea cifrei de control asociată unui număr ..... 5 puncte
- determinarea celui mai lung prefix ..... 15 puncte
- afișarea celui mai lung prefix/mesaj ..... 3 puncte

Program principal: ..... 3 puncte

- comunicare prin parametri: (signatura subalgorimilor și apelul corect) ..... 5 puncte
- lizibilitate:
  - comentarii ..... 2 puncte
  - indentare ..... 2 puncte
  - denumiri sugestive ..... 2 puncte

Concurs Mate-Info - 1 aprilie 2017

Proba scrisă la Informatică

VARIANTA I

```
// SUBIECTUL I.1

//determina cmmdc a 2 numere a si b
int cmmdc(int a, int b){
	if ((a == b) && (a == 0)){
		return 1;
	}
	if (a * b == 0){
		return a + b;
	}
	while (a != b){
		if (a > b)
			a -= b;
		else
			b -= a;
	}
	return a;
}

// calcularea numărului de schimbări de direcție a razei
int raza(int a, int b){
	int nrSchimb;
	int d = cmmdc(a, b);
	nrSchimb = b / d + a / d - 2;
	return nrSchimb;
}

// SUBIECTUL I.2

const int MAXSIZE = 100;
const int MAXCIF = 32;

// calcularea fortei unui numar
int calculForta(int nr){
	int forta = 0;
	do{
		nr &= nr - 1; // „și” logic intre nr si nr-1
		forta++;
	} while (nr);
	return forta;
}

//gruparea numerelor pe clase de forta
//grupurile se vor memora intr-un tablou bidimensional
//fiecare linie i corespunde grupului de forta i
//primul element de pe linia i reprezinta dimensiunea grupului de forta i
//urmatoarele elemente sunt numerele din sirul x cu forta i
void forte(int n, int x[], int &nrGr, int grupuri[][MAXSIZE]){
	for (int i = 1; i < MAXCIF; i++)  // inițializarea tabloului grupuri
		grupuri[i][0] = 0;
	nrGr = 0;
	for (int i = 0; i < n; i++){
		int forta = calculForta(x[i]);    // determinarea forței elementului x[i]
		if (grupuri[forta][0] == 0)
			nrGr++;
		int pos = grupuri[forta][0] + 1;
		grupuri[forta][pos] = x[i];
		grupuri[forta][0]++;
	}

}
```

```
// SUBIECTUL II
int fRec(int a, int b, int nr){
	if (b < nr)
		return fRec(b, a + b, nr) + 1; // (*)
	else
		return 0;
}



// SUBIECTUL III
const int MAX = 100;
const int NRMAXCIFRE = 10;

//citirea unui numar
void citireNumar(int &nr){
	cin >> nr;
}

//citirea elementelor unei matrici
void citireMatrice(int &m, int &n, int A[][MAX]){
	cin >> m >> n;
	for (int i = 0; i < m; i++){
		for (int j = 0; j < n; j++)
			cin >> A[i][j];
	}
}

//citirea numarului si a elementelor din matrice
void citireDate(int &nr, int &m, int &n, int A[][MAX]){
	citireNumar(nr);
	citireMatrice(m, n, A);
}

//calcularea cifrei de control a unui numar
int cifraControl(int x){
	while (x > 9){
		int y = x;
		int s = 0;
		while (y > 0){
			s += y % 10;
			y /= 10;
		}
		x = s;
	}
	return x;
}
```

```
//determinarea prefixului maxim
//matricea se parcurgere o singură dată și se construiește un vector de apariții
//pentru cifrele de control corespunzătoare elementelor din matrice.
//Se reține într - un vector cifrele numărului dat(nr).
//Se parcurge vectorul acestor cifre(începând cu cifra cea mai semnificativă) și
//se verifică apariția cifrelor în vectorul de apariții construit anterior
int prefixMaxCifre(int nr, int m, int n, int A[][MAX], int cifre[], int &nrCifre){
	bool aparitii[NRMAXCIFRE];
	int i = 0;
	for (i = 0; i < 10; i++)    // initializarea vectorului de frecvente
		aparitii[i] = 0;
	for (i = 0; i < m; i++){
		for (int j = 0; j < n; j++){
			aparitii[cifraControl(A[i][j])] = 1;     //  aparitia cifrei de control
		}                                    // corespunzatoare elementelui din matricea A
	}
	nrCifre = 0;         // numarul cifrelor numarului nr dat
	while (nr > 0){
		cifre[nrCifre++] = nr % 10; // elementele sirului cifrelor numarului nr dat
		nr /= 10;
	}

	i = nrCifre - 1;     // parcurgem sirul cifrelor
	while ((i >= 0) && (aparitii[cifre[i]])) // cifra de control = cu cifra curenta din nr
		i--;
	return nrCifre - i - 1;
}


//se afiseaza prefixul maxim de lungime lung cu cifre din vectorul de cifre
void afisarePrefix(int lung, int cifre[], int nrCifre){
	if (lung == 0)
		cout << "nu exista prefix";
	for (int i = 0; (i < lung && i < nrCifre); i++)
		cout << cifre[nrCifre - i - 1];
	cout << endl;
}



int main(){

	int nr = -1;
	int m = -1;
	int n = -1;
	int A[MAX][MAX];
	citireDate(nr, m, n, A);

	int cifre[NRMAXCIFRE];
	int nrCifre = 0;
	int lung = prefixMaxCifre(nr, m, n, A, cifre, nrCifre);
	afisarePrefix(lung, cifre, nrCifre);

	return 0;
}
```