PROIECTAREA SI IMPLEMENTAREA ALGORITMILOR – LABORATOR 2

Tablouri unidimensionale

În limbajul Java lucrul cu tablouri unidimensionale (sau vectori) presupune parcurgerea a două etape: declarare și alocare de memorie.

1. Declarare: Tip_componente numetablou [];
   sau Tip_componente[] numetablou;

2. Alocare de memorie: numetablou = new Tip_componente [dimensiune];

Componentele tabloului unidimensional sunt:
numetablou[0],numetablou[1],…,numetablou[dimensiune-1]

Operațiile specifice cu vectori sunt aceleași ca în C++.

Problema rezolvata: Se dă un vector cu n componente numere naturale mai mici sau egale cu 10000, $1 \leq n \leq 100$. Afișați pe rânduri diferite componentele pare, respectiv impare.

Soluție:

```
import java.util.Scanner;

public class vector1 {

        public static void main(String[] args) {

        int n,i,x[];

       Scanner cin=new Scanner(System.in);

        n=cin.nextInt();

        x=new int [n+1];

       for(i=1;i<=n;i++) x[i]=cin.nextInt();

        for(i=1;i<=n;i++)

                if(x[i]%2==0) System.out.print(x[i]+" "); System.out.println();

       for(i=1;i<=n;i++)

                if(x[i]%2==1) System.out.print(x[i]+" "); } }
```

Tablouri bidimensionale

În limbajul Java lucrul cu tablouri bidimensionale presupune parcurgerea a două etape: declarare și alocare de memorie.

1. Declarare: Tip_componente numetablou[][];
   sau Tip_componente[][] numetablou;
2. Alocare de memorie: numetablou = new Tip_componente [dim1] [dim2];

Componentele tabloului unidimensional sunt : numetablou[i][j], $1 \leq i \leq dim1 - 1$, $1 \leq j \leq dim2 - 1$. Operațiile specifice cu tablouri bidimensionale sunt aceleași ca în C++.

Problema rezolvata: Se dă un tablou pătratic de dimensiune n cu componente numere naturale mai mici sau egale cu 10000, $1 \leq n \leq 100$. Ordonați crescător fiecare linie și apoi afișați tabloul.

Soluție:

```
import java.util.Scanner;

public class P2 {

public static void main(String[] args) {

        Scanner cin = new Scanner (System.in);

         int a[][],n,i,j,k,aux;

        n=cin.nextInt();

        a= new int[n+1][n+1];

        for (i=1;i<=n;i++)

        for (j=1;j<=n;j++) a[i][j]=cin.nextInt();

        for (i=1;i<=n;i++){

                for(j=1;j<=n-1;j++)

                for(k=j+1;k<=n;k++)

                        if (a[i][j]>a[i][k]){

                        aux=a[i][j];

                        a[i][j]=a[i][k];

                        a[i][k]=aux; } }
```

```
for (i=1;i<=n;i++){

for (j=1;j<=n;j++)

System.out.print(a[i][j]+ " ");

System.out.println(); } } }
```

Programe Java cu o singură clasă. Date membru și metode

Un program în limbajul Java care conține o singură clasă (clasa principală) are următoarea formă generală:

```
public class nume{

    static tip lista date membru;

    static tip numefunctie(parametrii){ … … }

    public static void main(String[] args){ … … } }
```

Datele membru sunt variabile ce pot fi utilizate în toate metodele (funcțiile) clasei.

Observații:

1. Apelul metodelor este numai prin valoare, acestea putând fi utilizate ca în limbajul C++.

2. Ordinea de scriere a metodelor și datelor membru nu are importanță.

3. Returnarea unei valori într-o metodă se realizează ca în C++, folosind instrucțiunea return.

4. Se pot defini într-o clasă și metode recursive.

5. Pentru o clasă datele membru sunt precum variabilele globale într-un program C++.

Problema rezolvata cu metode nerecursive: Se dă n număr natural cu cel mult 9 cifre. Afișați numerele palindrom mai mici sau egale cu n.

Soluție: Vom crea o clasă cu n - dată membru și două metode: main și o o metodă pentru verificarea condiției de număr palindrom.

```
import java.util.Scanner;
public class program1 {
static int n;
static Scanner cin=new Scanner(System.in);
static boolean palindrom(int k) {
        int aux,c,inv;
        inv=0;
        aux=k;
                while (aux>0) {
                c=aux%10;
                 inv=inv*10+c;
                aux/=10; }
        if (inv==k) return true;
        return false; }
public static void main(String[] args) {
System.out.print("n=");
n=cin.nextInt();
int i;
for (i=0;i<=n;i++) if (palindrom(i)) System.out.print(i+" "); } }
```

Problema rezolvata folosind metode recursive: Se dă n număr natural mai mic sau egal cu 15. Afișați valoarea sumei 1! + 2! + …+ n!.

Soluție:

```
import java.util.Scanner;

public class program6 {

        static int n;

        static int factorial(int k){

         if (k==1) return 1;

        return factorial(k-1)*k; }

        public static void main(String[] args) {

        int i,s=0;

        Scanner cin = new Scanner(System.in);

        System.out.print("n=");

        n=cin.nextInt();

        for(i=1;i<=n;i++) s+=factorial(i);

        System.out.print(s); } }
```

Probleme propuse:

1. Se dă n număr natural cu cel mult 5 cifre. Afișați numerele prime din intervalul [n, 2n].
2. Se dau a și b numere naturale cu maxim 9 cifre fiecare. Pentru fiecare număr a, b, afișați suma și produsul cifrelor nenule.
3. Se dă un vector cu n componente numere naturale. Câte componente termeni din șirul lui Fibonacci (1, 1, 2, 3, 5, 8, 13, …) sunt în vector?
4. Se dă un tablou pătratic de dimensiune n cu componente numere naturale mai mici sau egale cu 10000, $1 \leq n \leq 100$. Afișați pentru fiecare linie cel mai mare, respectiv cel mai mic număr separate prin câte un spațiu.
5. Se dă n număr natural mai mic sau egal cu 10. Afișați toate numerele cu exact n cifre, toate impare, astfel încât să nu existe două cifre egale una lângă alta.
   Exp: Intrare 4 Ieșire 1313 1315 1317
6. Se dă n număr natural mai mic sau egal cu 20. Afișați toate modalitățile de scriere a lui n ca sumă de numere impare.

Referinte: BAZELE PROGRAMARII - JAVA DUPA C++ Autor: Doru Anastasiu Popescu