# PROIECT NR.2:ELEMENTE DE COMBINATORICĂ

A elaborat: Dmitri COJOCARI, gr. IS11Z

A verificat: Vitalie ȚÎCĂU, lector universitar

## 1. Formularea problemei

1. De elaborat un program ce va calcula următoarele elemente de combinatorică

1. permutări;
2. aranjamente;
3. combinări;
4. permutări cu repetiții;
5. aranjamente cu repetiții
6. combinări cu repetiții

## 2. Studiu preliminar

Pentru implementarea elementelor enumerate mai sus, trebuie utilizat un subprogram ce va calcula factorialul. Întrucât factorialul este definit doar penru numere naturale, vom utiliza tipul unsigned long long pentru acest subprogram.

Permutările determină numărul de ordonări posibile ale elementelor unei mulțimi (utilizând toatele elementele mulțimii).

Aranjamentele din n luate câte k reprezintă totalitatea mulțimilor ordonate alcătuite din k elemente ale unei mulțimi A(ordinea elementelor contează).

Combinările indică numărul de submulțimi neordonate a unei mulțimi(ordinea elementelor nu contează).

De asemena, am implementat funcția proprie de ridicare la putere, deoarece funcția standard pow în unele cazuri duce la erori de aproximare.

## 3. Programul

#include <iostream>

#include <stdio.h>

#include <string.h>

using **namespace** std;

**unsigned** **long** **long** Factorial(**unsigned**);

**unsigned** **long** **long** Permutari(**unsigned**);

**unsigned** **long** **long** Aranjamente(**unsigned**, **unsigned**);

**unsigned** **long** **long** Combinari(**unsigned**, **unsigned**);

**unsigned** **long** **long** Power(**unsigned**, **unsigned**);

**unsigned** **long** **long** PowerRec(**unsigned**, **unsigned**);

**unsigned** **long** **long** PermutariCuRepetitii(**unsigned**);

**unsigned** **long** **long** AranjamenteCuRepetitii(**unsigned**, **unsigned**);

**unsigned** **long** **long** CombinariCuRepetitii(**unsigned**, **unsigned**);

**unsigned** **long** **long** PermutariCuRepetitii2(**unsigned**, **unsigned**[]);

**int** main()

{

    cout << "Proiect nr.2" << endl;

    cout << "Tema: Elemente de combinatorica" << endl;

    cout << "Elaborat de Cojocari Dmitri" << endl;

**unsigned** a[10] = {1, 3, 2, 0, 0};

**char** sir[100] = "Pr(", Sir[100];

**unsigned** n = 6, k = 2, m;

    cout << "Numarul total de elemente al multimii: ";

    cin >> n;

    do

    {

        cout << "Numarul de elemente al grupului (cate elemente luam in grup): ";

        cin >> k;

    } while (k > n || k < 0);

    for (**int** i = 0; i < n; i++)

    {

        a[i] = 0;

    }

    cout << "Introduceti " << n << " elemente: " << endl;

    for (**int** i = 0; i < n; i++)

    {

        cin >> m;

        a[m - 1]++;

    }

    cout << n << "^" << k << " = " << PowerRec(n, k) << endl;

    cout << n << "! = " << Factorial(n) << endl;

    cout << "P(" << n << ")"

         << " = " << Permutari(n) << endl;

    cout << "A(" << n << "," << k << ")"

         << " = " << Aranjamente(n, k) << endl;

    cout << "C(" << n << "," << k << ")"

         << " = " << Combinari(n, k) << endl;

    cout << "Pr(" << n << ")"

         << " = " << PermutariCuRepetitii(n) << endl;

    cout << "Ar(" << n << ")"

         << " = " << AranjamenteCuRepetitii(n, k) << endl;

    cout << "Cr(" << n << ")"

         << " = " << CombinariCuRepetitii(n, k) << endl;

**int** d = 0;

    while (a[d] == 0)

    {

        d++;

    }

    sprintf(Sir, "%s%u", sir, a[d]);

    strcpy(sir, Sir);

    for (**int** i = d + 1; i < n; i++)

    {

        if (a[i] != 0)

        {

            sprintf(Sir, "%s%c%u", sir, ',', a[i]);

        }

        strcpy(sir, Sir);

    }

    m = strlen(sir);

    strcpy(&sir[1], &sir[m]);

*//sir[strlen(sir) - 1] = '\0';*

    cout << Sir << ") = " << PermutariCuRepetitii2(n, a);

    return 0;

}

**unsigned** **long** **long** Factorial(**unsigned** m)

{

**unsigned** **long** **long** f = 1;

    for (**int** i = 2; i <= m; i++)

    {

        f \*= i;

    }

    return f;

}

**unsigned** **long** **long** Permutari(**unsigned** n)

{

    return Factorial(n);

}

**unsigned** **long** **long** Aranjamente(**unsigned** n, **unsigned** k)

{

    return (Factorial(n) / Factorial(n - k));

}

**unsigned** **long** **long** Combinari(**unsigned** n, **unsigned** k)

{

    return (Factorial(n) / (Factorial(k) \* Factorial(n - k)));

}

**unsigned** **long** **long** Power(**unsigned** n, **unsigned** k)

{

**unsigned** **long** **long** p = 1;

    for (**int** i = 1; i <= k; i++)

    {

        p = p \* n;

    }

    return p;

}

**unsigned** **long** **long** PowerRec(**unsigned** n, **unsigned** k)

{

**unsigned** **long** **long** p = 1;

    if (k <= 0)

        return 1;

    return (PowerRec(n, k - 1) \* n);

}

**unsigned** **long** **long** PermutariCuRepetitii(**unsigned** n)

{

    return Power(n, n);

}

**unsigned** **long** **long** AranjamenteCuRepetitii(**unsigned** n, **unsigned** k)

{

    return Power(n, k);

}

**unsigned** **long** **long** CombinariCuRepetitii(**unsigned** n, **unsigned** k)

{

    return (Factorial(n + k - 1) / (Factorial(k) \* Factorial(n - 1)));

}

**unsigned** **long** **long** PermutariCuRepetitii2(**unsigned** size, **unsigned** a[])

{

*// Exemplu: avem 3 cifre de 2, 2 cifre de 3, 1 cifra de 1. Ce numere putem alcatui?*

*// 222331*

*// 122233, 122323, 122332, 123223 ...*

*// P2(2,3,1) = (2+3+1)! / (2!\*3!\*1!)*

**unsigned** **long** **long** p = 1;

**unsigned** s = 0;

    for (**int** i = 0; i < size; i++)

    {

        s += a[i];

        p \*= Factorial(a[i]);

    }

    return Factorial(s) / p;

}

## 4. Rezultatele si analiza lor

Proiect nr.2

Tema: Elemente de combinatorica

Elaborat de Cojocari Dmitri

Numarul total de elemente al multimii: 6

Numarul de elemente al grupului (cate elemente luam in grup): 3

Introduceti 6 elemente:

2

2

2

3

3

1

6^3 = 216

6! = 720

P(6) = 720

A(6,3) = 120

C(6,3) = 20

Pr(6) = 46656

Ar(6) = 216

Cr(6) = 56

Pr(1,3,2) = 60

Rezultatele corespund rezultatelor obținute din formulele cunoscute.

## 5. Concluzii

În urma realizării prezentei lucrări, am observat că:

* Permutările reprezintă numărul total de ordonări a unei mulțimi utilizând toate elementele ale acesteia (P(n) = n!);
* Aranjamentele sunt utilizate atunci când ordinea elementelor contează;
* Combinările se folosesc atunci când ordinea elementelor nu contează
* În cazul elementelor cu repetiții, se admite repetiția elementelor din mulțimea inițială.