



matek, fizika  
programozás  
*könnyedén*

programozási tételek C++ nyelven

```
#include <iostream>
using namespace std;
void writeArray(int aa[], int size)
{
    int i = 0;
    while (i < size)
    {
        aa[i] = 1 + rand() % 10;
        i++;
    }
}
void readArray(int aa[], int size)
{
    int i = 0;
    while (i < size)
    {
        cout << aa[i] << endl;
        i++;
    }
    cout << "-----" << endl;
}

//1. összegzés tétel
int summationOfElements(int aa[], int size)
{
    int i = 0;
    int sum = 0;
    while (i < size)
    {
        sum += aa[i];
        i++;
    }
    return sum;
}

//2. csere tétel
void replacamentTheorem(int& x, int& y)
{
    int temp = 0;
    temp = x;
    x = y;
    y = temp;
}
```

```

bool P_property(int x)
{
    //if (x%3==0)
    //{
        // return true;
    //}
    //else
    //{
        // return false;
    //}
    return x % 3 == 0;
}

//3. eldöntés tétel (1)
void decisionMaking1(int aa[], int size)
{
    int i = 0;
    while (i < size)
    {
        if (P_property(aa[i]))
        {
            cout << "There is an element with P property in the array!" <<
endl;
        }
        else
        {
            cout << "There is NO element with P property in the array!" <<
endl;
        }
        i++;
    }
    cout << "----" << endl;
}

//3. eldöntés tétel (2)
void decisionMaking2(int aa[], int size)
{
    int i = 0;
    while (i < size && !P_property(aa[i]))
    {
        i++;
    }
    if (i < size)
    {
        cout << "There is an element with P property!" << endl;
    }
    else
    {
        cout << "There is NO element with P property!" << endl;
    }
    cout << "----" << endl;
}

```

```

//4. feltételes összegzés tétel
int conditionalSum(int aa[], int size)
{
    int sum = 0;
    int i = 0;
    while (i < size)
    {
        if (P_property(aa[i]))
        {
            sum += aa[i];
        }
        i++;
    }
    return sum;
}

```

```

//5. feltételes megszámlálás tétel
int conditionalCount(int aa[], int size)
{
    int count = 0;
    int i = 0;
    while (i < size)
    {
        if (P_property(aa[i]))
        {
            count++;
        }
        i++;
    }
    return count;
}

```

```

//6. feltételes kiválasztás, kiválasztás tétel
int conditionalSelect(int aa[], int size)
{
    int index = 0;
    int i = 0;
    while (i < size)
    {
        if (P_property(aa[i]))
        {
            index = i;
            return index;
        }
        i++;
    }
}

```

```

//7. maximum kiválasztás tétel
int selectMax(int aa[], int size)
{
    int max_i = 0;
    int i = 0;
    while (i < size)
    {
        if (aa[max_i] < aa[i])
        {
            max_i = i;
        }
        i++;
    }
    return max_i;
}

```

```

//8. minimum kiválasztás tétel
int selectMin(int aa[], int size)
{
    int min_i = 0;
    int i = 0;
    while (i < size)
    {
        if (aa[min_i] > aa[i])
        {
            min_i = i;
        }
        i++;
    }
    return min_i;
}

```

```

//9. lineáris keresés tétel
void linearSearch(int aa[], int size)
{
    int i = 0;
    while (i < size && !P_property(aa[i]))
    {
        i++;
    }
    if (i < size)
    {
        cout << "There is an element with P property, with index of: " << i + 1 << endl;
    }
    else
    {
        cout << "There is NO element with P property!" << endl;
    }
    cout << "----" << endl;
}

```

```

//10. egyszerű cserés rendezés
void replacamentSort(int aa[], int size)
{
    cout << "Egyszerű cserés rendezés!" << endl;
}

```

```

cout << "Array before sorting: " << endl;
int i = 0;
while (i < size)
{
    cout << aa[i] << endl;
    i++;
}
cout << "-----" << endl;
i = 0;
int j = 0;
while (i < size - 1)
{
    j = 0;
    while (j < size - 1)
    {
        if (aa[j] > aa[j + 1])
        {
            int seged = aa[j];
            aa[j] = aa[j + 1];
            aa[j + 1] = seged;
        }
        j++;
    }
    i++;
}
cout << "Array after sorting: " << endl;
i = 0;
while (i < size)
{
    cout << aa[i] << endl;
    i++;
}
}

```

```

//10. buborék rendezés
void bubbleSort(int aa[], int size)
{
    cout << "Buborék rendezés!" << endl;
}

```

```

cout << "Array before sorting: " << endl;
int i = 0;
while (i < size)
{
    cout << aa[i] << endl;
    i++;
}
cout << "----" << endl;
i = size - 1;
int j;
while (i > 0)
{
    j = 0;
    while (j < i)
    {
        if (aa[j] > aa[j + 1])
        {
            int seged = aa[j];
            aa[j] = aa[j + 1];
            aa[j + 1] = seged;
        }
        j++;
    }
    i--;
}
cout << "Array after sorting: " << endl;
i = 0;
while (i < size)
{
    cout << aa[i] << endl;
    i++;
}
}

```

```

//11. max, min kiválasztásos rendezés
void maxSelectSort(int aa[], int size)
{
    cout << "Max. kiválasztásos rendezés!" << endl;
}

```

```

cout << "Array before sorting: " << endl;
int i = 0;
while (i < size)
{
    cout << aa[i] << endl;
    i++;
}
cout << "-----" << endl;
int max_index = 0;
i = size - 1;
int j = 0;
int seged = 0;
while (i >= 0)
{
    while (j <= i)
    {
        if (aa[j] > aa[max_index])
        {
            max_index = j;
        }
        j++;
    }
    seged = aa[i];
    aa[i] = aa[max_index];
    aa[max_index] = seged;
    i--;
    max_index = 0;
    j = 0;
}
i = 0;
while (i < size)
{
    cout << aa[i] << endl;
    i++;
}
}

```

```

//12. kiválogatás
void assortment(int aa[], int size)
{
    cout << "assortment: " << endl;
}

```

```

int* p = new int[size];
int numberOfAssortedElements = 0;
int j = 0;
int i = 0;
while (i < size)
{
    if (P_property(aa[i]))
    {
        p[j] = aa[i];
        j++;
    }
    else
    {
        p[j] = 0;
    }
    i++;
}
if (j == 0)
{
    cout << "There is no element with P_property! " << endl;
}
numberOfAssortedElements = j;
i = 0;
while (i < numberOfAssortedElements)
{
    cout << "The selected elements are: " << p[i] << endl;
    i++;
}
delete[] p;
}

```

```

int main()
{
    setlocale(LC_ALL, "hun");
    const int size = 10;
    int aa[size] = { 0 };
}

```



```

writeArray(aa, size);
readArray(aa, size);
int sum = summationOfElements(aa, size);
cout << "összeg: " << sum << endl;
cout << "----" << endl;
int x = 10;
int y = 20;
cout << "x: " << x << ", y: " << y << endl;
replacamentTheorem(x, y);
cout << "x: " << x << ", y: " << y << endl;
cout << "----" << endl;
decisionMaking1(aa, size);
decisionMaking2(aa, size);
int cSum = conditionalSum(aa, size);
cout << "feltételes összeg: " << cSum << endl;
int cCount = conditionalCount(aa, size);
cout << "feltételes megszámlálás: " << cCount << endl;
int cIndex = conditionalSelect(aa, size);
cout << "feltételesen kiválasztott elem indexe: " << cIndex + 1 << endl; //human számozás
int indexOfMaxElement = selectMax(aa, size);
cout << "A legnagyobb elem indexe: " << indexOfMaxElement + 1 << endl; //human számozás
int indexOfMinElement = selectMin(aa, size);
cout << "A legkisebb elem indexe: " << indexOfMinElement + 1 << endl; //human számozás
linearSearch(aa, size);
replacamentSort(aa, size);
writeArray(aa, size);
bubbleSort(aa, size);
writeArray(aa, size);
maxSelectSort(aa, size);
assortment(aa, size);
system("pause");
return 0;
}

```