



28TECH
Become A Better Developer



MẢNG MỘT CHIỀU





/PROBLEM

Giả sử bạn cần tính trọng lượng của 1 đàn gà lên đến **hàng nghìn con**. Vậy việc lưu trữ trọng lượng của hàng nghìn con gà này sẽ phải xử lý ra sao?





1. Định nghĩa và tính chất của mảng 1 chiều:



Là một **cấu trúc dữ liệu** gồm nhiều phần tử có cùng kiểu dữ liệu, được lưu trữ ở **các ô nhớ liên tiếp** nhau trong bộ nhớ.



Được sử dụng khi bạn cần lưu trữ một số lượng lớn các phần tử có **cùng kiểu dữ liệu**.



Mảng 1 chiều **đơn giản, dễ hiểu và được sử dụng rất nhiều** trong mọi ngôn ngữ lập trình.



2. Khai báo mảng:

CÚ PHÁP & VÍ DỤ:

```
Data_type[] array_name = new Data_type [Number_of_element];
```

| Khai báo | Ý nghĩa |
|--|--|
| <code>int[] a = new int[100];</code> | Mảng số nguyên int a có 100 phần tử |
| <code>float[] b = new float[1000];</code> | Mảng số thực float b có 1000 phần tử |
| <code>double[] diem = new double[10];</code> | Mảng số thực double diem có 10 phần tử |
| <code>char[] ten = new char[50];</code> | Mảng kí tự char ten có 50 phần tử |

2. Khai báo mảng:

- Khai báo mảng a có 3 phần tử là số nguyên, các giá trị của a sẽ là giá trị mặc định là 0.

```
int[] a = new int[3];
```

| | | |
|-----|-------|---|
| 100 | 20221 | 0 |
|-----|-------|---|

- Khai báo mảng a có 3 phần tử là số nguyên, gán lần lượt các phần tử trong mảng a là 1, 2, 3:

```
int[3] a = {1, 2, 3};
```

| | | |
|---|---|---|
| 1 | 2 | 3 |
|---|---|---|



3. Truy cập phần tử và duyệt mảng:



Các phần tử trong mảng được truy cập thông qua chỉ số, chỉ số của mảng được đánh từ 0 và kết thúc bởi $n - 1$ với n là số lượng phần tử của mảng.

CÚ PHÁP:

```
array_name[index];
```

EXAMPLE

```
int[6] a = {3, 8, 9, 1, 7, 4};
```

| | | | | | | |
|-------|---|---|---|---|---|---|
| Array | 3 | 8 | 9 | 1 | 7 | 4 |
| Index | 0 | 1 | 2 | 3 | 4 | 5 |





3. Truy cập phần tử và duyệt mảng:

Duyệt mảng thông qua chỉ số

```
public class Main {  
    public static void main(String[] args){  
        Scanner sc = new Scanner(System.in);  
        int n = sc.nextInt();  
        int[] a = new int[n];  
        for(int i = 0; i < n; i++){  
            a[i] = sc.nextInt();  
        }  
        for(int i = 0; i < n; i++){  
            System.out.print(a[i] + " ");  
        }  
    }  
}
```

Duyệt mảng bằng for each

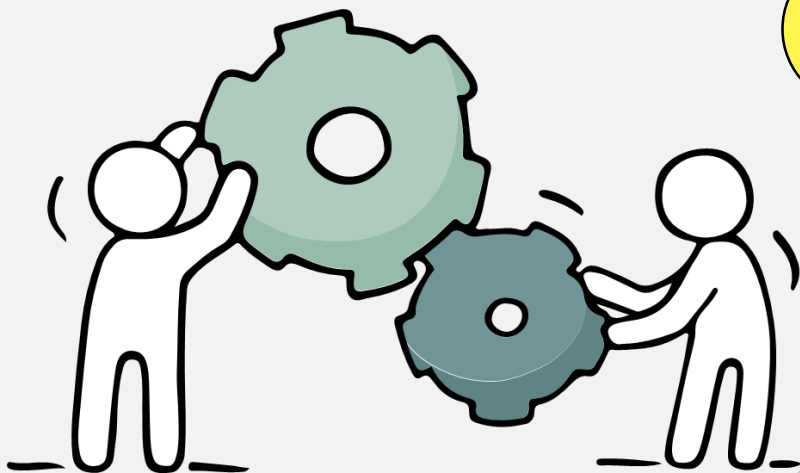
```
public class Main {  
    public static void main(String[] args){  
        Scanner sc = new Scanner(System.in);  
        int n = sc.nextInt();  
        int[] a = new int[n];  
        for(int i = 0; i < n; i++){  
            a[i] = sc.nextInt();  
        }  
        for(int x : a){  
            System.out.print(x + " ");  
        }  
    }  
}
```

Chú ý: Khi khai báo kích thước của mảng hãy chú ý tới giới hạn số lượng phần tử tối đa của đầu bài.



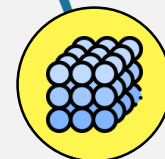


4. Mảng làm tham số của hàm:



Khi mảng làm tham số của hàm, những thay đổi trong hàm sẽ làm thay đổi tới mảng mảng được truyền vào.

Chú ý: Khi xây dựng hàm có tham số là mảng, cần phải kèm theo 1 tham số nữa là số lượng phần tử trong mảng





4. Mảng làm tham số của hàm:

Gấp đôi mọi phần tử trong mảng

```
public class Main {  
    public static void nhanDoi(int a[], int n){  
        for(int i = 0; i < n; i++){  
            a[i] *= 2;  
        }  
    }  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        int[] a = {1, 2, 3, 4};  
        nhanDoi(a, 4);  
        for(int x : a){  
            System.out.print(x + " ");  
        }  
    }  
}
```

OUTPUT

2 4 6 8





4. Mảng làm tham số của hàm:

Tính tổng các phần tử trong mảng

```
public class Main {  
    public static int tong(int a[], int n){  
        int sum = 0;  
        for(int i = 0; i < n; i++){  
            sum += a[i];  
        }  
        return sum;  
    }  
  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        int[] a = {1, 2, 3, 4};  
        System.out.println(tong(a, 4));  
    }  
}
```

OUTPUT

10



5. Ưu và nhược điểm của mảng:



- Đơn giản, dễ hiểu và dễ sử dụng
- Truy cập vào phần tử trong mảng nhanh chóng thông qua chỉ số.
- Dễ dàng khai báo một loạt các phần tử cùng kiểu dữ liệu.
- Dễ dàng duyệt mọi phần tử trong mảng bằng một vòng lặp duy nhất.



- Kích thước của mảng là cố định, bạn không thể mở rộng cũng như thu hẹp mảng một khi nó đã được khai báo.
- Việc chèn và xóa phần tử trong mảng là khó khăn.



6. Một số bài toán cơ bản cần lưu ý:

a. Tìm phần tử nhỏ nhất và lớn nhất trong mảng:

Cách 1:

```
public class Main {  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        int n = sc.nextInt();  
        int[] a = new int[n];  
        for(int i = 0; i < n; i++){  
            a[i] = sc.nextInt();  
        }  
        int maxElement = a[0], minElement = a[0];  
        for(int i = 1; i < n; i++){  
            if(a[i] > maxElement)  
                maxElement = a[i];  
            if(a[i] < minElement)  
                minElement = a[i];  
        }  
        System.out.println(maxElement + " " + minElement);  
    }  
}
```





6. Một số bài toán cơ bản cần lưu ý:

a. Tìm phần tử nhỏ nhất và lớn nhất trong mảng:

Cách 2:

```
public class Main {  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        int n = sc.nextInt();  
        int[] a = new int[n];  
        for(int i = 0; i < n; i++){  
            a[i] = sc.nextInt();  
        }  
        int maxElement = a[0], minElement = a[0];  
        for(int i = 1; i < n; i++){  
            maxElement = Math.max(a[i], maxElement);  
            minElement = Math.min(a[i], minElement);  
        }  
        System.out.println(maxElement + " " + minElement);  
    }  
}
```





6. Một số bài toán cơ bản cần lưu ý:

b. Liệt kê hoặc đếm các phần tử trong mảng thỏa yêu cầu:

VD: Chương trình liệt kê số nguyên tố

```
public class Main {  
    public static boolean nt(int n) {  
        for (int i = 2; i <= Math.sqrt(n); i++)  
            if (n % i == 0)  
                return false;  
        return n > 1;  
    }  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        int n = sc.nextInt();  
        int[] a = new int[n];  
        for (int i = 0; i < n; i++)  
            a[i] = sc.nextInt();  
        for (int i = 0; i < n; i++)  
            if (nt(a[i])) {  
                System.out.print(a[i] + " ");  
            }  
    }  
}
```

Ngoài số nguyên tố thì còn liệt kê số chính phương, số Fibonacci trong mảng, đếm số chẵn, lẻ...





6. Một số bài toán cơ bản cần lưu ý:

c. Các bài toán liên quan tới cặp số trong mảng:

VD: Đếm số cặp trong mảng có tổng bằng K

```
public class Main {  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        int n = sc.nextInt(), k = sc.nextInt();  
        int[] a = new int[n];  
        for (int i = 0; i < n; i++)  
            a[i] = sc.nextInt();  
        int ans = 0;  
        for(int i = 0; i < n; i++)  
            for(int j = i + 1; j < n; j++)  
                if(a[i] + a[j] == k)  
                    ++ans;  
        System.out.println(ans);  
    }  
}
```

- Khi bạn cần xét mọi cặp 2 phần tử trong mảng thì bạn cần 2 vòng for lồng nhau.
- Tổng quát nếu bạn cần xét mọi cặp k phần tử thì bạn cần k vòng for lồng nhau





6. Một số bài toán cơ bản cần lưu ý:

d. Sắp xếp các phần tử trong mảng:

Sắp xếp tăng dần:

```
public class Main {  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        int n = sc.nextInt();  
        int[] a = new int[n];  
        for (int i = 0; i < n; i++) {  
            a[i] = sc.nextInt();  
        }  
        Arrays.sort(a);  
        for(int x : a){  
            System.out.println(x + " ");  
        }  
    }  
}
```

INPUT

5
1 3 2 5 4

OUTPUT

1 2 3 4 5





6. Một số bài toán cơ bản cần lưu ý:

d. Sắp xếp các phần tử trong mảng:

Sắp xếp giảm dần:

```
public class Main {  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        int n = sc.nextInt();  
        Integer[] a = new Integer[n];  
        for (int i = 0; i < n; i++) {  
            a[i] = sc.nextInt();  
        }  
        Arrays.sort(a, Collections.reverseOrder());  
        for(int x : a){  
            System.out.print(x + " ");  
        }  
    }  
}
```

INPUT

5
1 3 2 5 4

OUTPUT

5 4 3 2 1





6. Một số bài toán cơ bản cần lưu ý:

e. Tìm kiếm hoặc đếm số lần xuất hiện của 1 phần tử nào đó trong mảng:

Tìm kiếm sự xuất hiện của phần tử X trong mảng

```
public class Main {  
    public static boolean timKiem(int a[], int n, int x){  
        for(int i = 0; i < n; i++){  
            if(a[i] == x)  
                return true;  
        }  
        return false;  
    }  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        int n = sc.nextInt(), x = sc.nextInt();  
        int[] a = new int[n];  
        for (int i = 0; i < n; i++){  
            a[i] = sc.nextInt();  
        }  
        System.out.println(timKiem(a, n, x));  
    }  
}
```





6. Một số bài toán cơ bản cần lưu ý:

e. Tìm kiếm hoặc đếm số lần xuất hiện của 1 phần tử nào đó trong mảng:

Đếm số lần xuất hiện của X trong mảng

```
public class Main {  
    public static int count(int a[], int n, int x) {  
        int res = 0;  
        for (int i = 0; i < n; i++)  
            if (a[i] == x)  
                ++res;  
        return res;  
    }  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        int n = sc.nextInt(), x = sc.nextInt();  
        int[] a = new int[n];  
        for (int i = 0; i < n; i++)  
            a[i] = sc.nextInt();  
        System.out.println(count(a, n, x));  
    }  
}
```

