09 - 一维数组

C++ 程序设计基础

SOJ 信息学竞赛教练组

2024 年 6 月 5 日

目录

多重循环

- 一重循环

多重循环

- 一重循环
- 二重循环

多重循环

- 一重循环
- 二重循环
  - 在一重循环的循环体中再写一个循环

多重循环

- 一重循环
- 二重循环
  - 在一重循环的循环体中再写一个循环
- 三重循环

多重循环

- 一重循环
- 二重循环
  - 在一重循环的循环体中再写一个循环
- 三重循环
- ...

引入

- 使用循环的时候，都是一边输入数据一边处理，并没有将所有的数据都存储起来

引入

- 使用循环的时候，都是一边输入数据一边处理，并没有将所有的数据都存储起来
- 如果需要将循环处理的数据重新输出一遍，则没有办法实现

思考

当变量很多且需要记录他们的值时，该怎么办?

目录

数组

- 数组的概念
  - 数组是用于存储多个同类型数据的结构
  - 多个数据存放在一片连续的内存空间中

```
int a[8];
```

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
71	80	62	91	99	82	43	53

数组

- 数组的概念
  - 数组是用于存储多个同类型数据的结构
  - 多个数据存放在一片连续的内存空间中
- 数组的优势
  - 代码简洁
  - 通用、易维护

```
int a[8];
```

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
71	80	62	91	99	82	43	53

数组

- 数组的声明
  - **元素类型　数组名 [数组大小];**
  - `int a[8]; // 定义了可以存放 8 个整数的数组 a`

`int a[8];`

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
71	80	62	91	99	82	43	53

数组

- 数组的声明
  - **元素类型　数组名 [数组大小];**
  - `int a[8]; // 定义了可以存放 8 个整数的数组 a`
  - 数组中的基本单元叫作“元素”
  - 数组大小定义为可能存放元素数量的最大值（可适当再大一些）

`int a[8];`

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
71	80	62	91	99	82	43	53

数组

- 数组的声明
  - **元素类型　数组名 [数组大小];**
  - `int a[8]; // 定义了可以存放 8 个整数的数组 a`
  - 数组中的基本单元叫作“元素”
  - 数组大小定义为可能存放元素数量的最大值（可适当再大一些）
  - 建议声明数组为全局变量

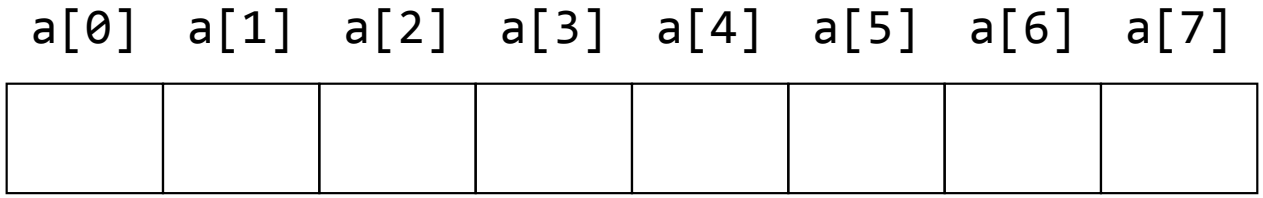
`int a[8];`

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
71	80	62	91	99	82	43	53

数组

- 数组的初始化
  - 可以通过一个赋值符号和花括号，将花括号中给定的数值序列，从第 0 位开始，依次赋值给数组中的每一个元素
    - `int a[8] = {71, 80, 62, 91, 99, 82, 43, 53};`

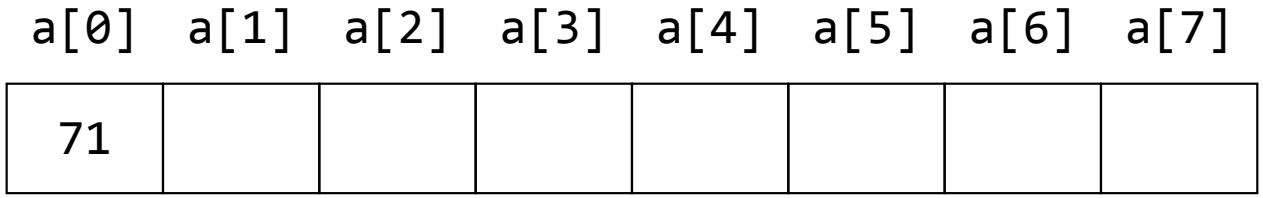
`int a[8];`



数组

- 数组的初始化
  - 可以通过一个赋值符号和花括号，将花括号中给定的数值序列，从第 0 位开始，依次赋值给数组中的每一个元素
    - `int a[8] = {71, 80, 62, 91, 99, 82, 43, 53};`

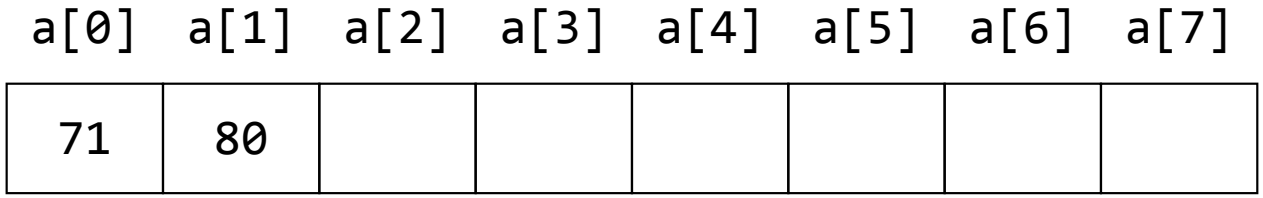
`int a[8];`



数组

- 数组的初始化
  - 可以通过一个赋值符号和花括号，将花括号中给定的数值序列，从第 0 位开始，依次赋值给数组中的每一个元素
    - `int a[8] = {71, 80, 62, 91, 99, 82, 43, 53};`

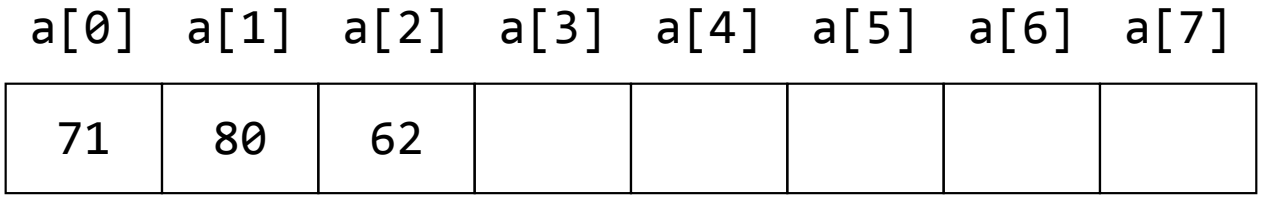
`int a[8];`



数组

- 数组的初始化
  - 可以通过一个赋值符号和花括号，将花括号中给定的数值序列，从第 0 位开始，依次赋值给数组中的每一个元素
    - `int a[8] = {71, 80, 62, 91, 99, 82, 43, 53};`

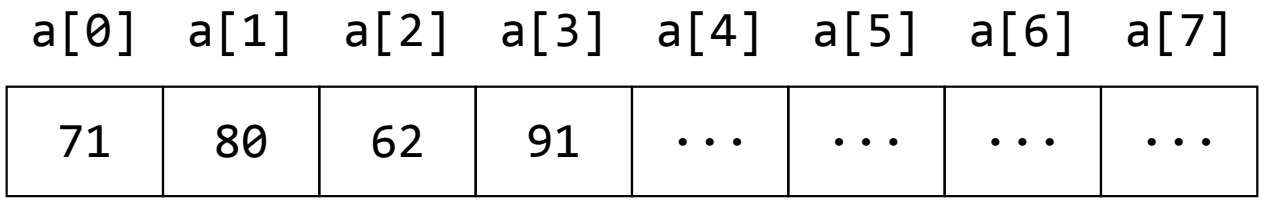
`int a[8];`



数组

- 数组的初始化
  - 可以通过一个赋值符号和花括号，将花括号中给定的数值序列，从第 0 位开始，依次赋值给数组中的每一个元素
    - `int a[8] = {71, 80, 62, 91, 99, 82, 43, 53};`

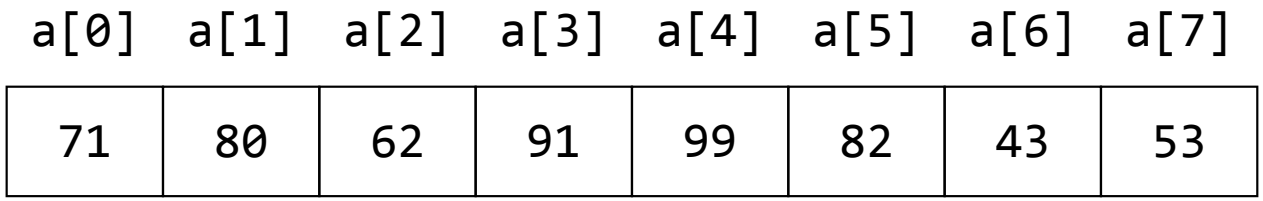
`int a[8];`



数组

- 数组的初始化
  - 可以通过一个赋值符号和花括号，将花括号中给定的数值序列，从第 0 位开始，依次赋值给数组中的每一个元素
    - `int a[8] = {71, 80, 62, 91, 99, 82, 43, 53};`

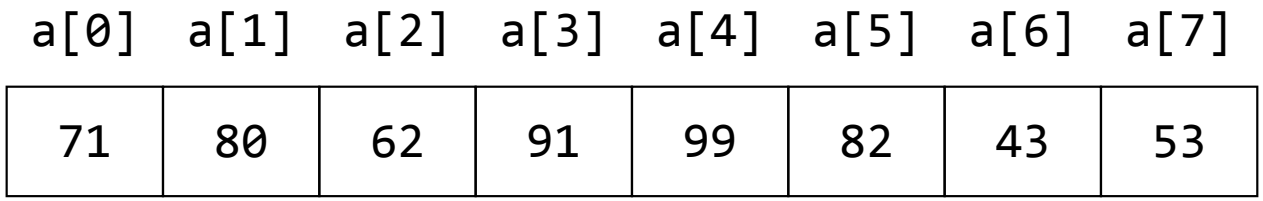
`int a[8];`



数组

- 数组的初始化
  - 可以通过一个赋值符号和花括号，将花括号中给定的数值序列，从第 0 位开始，依次赋值给数组中的每一个元素
    - `int a[8] = {71, 80, 62, 91, 99, 82, 43, 53};`

`int a[8];`



数组

- 数组的初始化
  - 可以通过一个赋值符号和花括号，将花括号中给定的数值序列，从第 0 位开始，依次赋值给数组中的每一个元素
    - `int a[8] = {71, 80, 62, 91, 99, 82, 43, 53};`
  - 数组大小可以指定或留空，会按照需要的最小空间申请内存
    - `double b[] = {1.0, 2.7, 3.14, 9.8};`

`int a[8];`

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
71	80	62	91	99	82	43	53

数组

- 数组元素的访问
  - 通过 **数组名 [下标]** 访问存储在数组中某一位置的值
    - 下标是指在元素在数组中的位置，从 0 开始
    - 下标的范围：$0 \sim L-1$ (L 为数组大小)

```
int a[8];
```

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
71	80	62	91	99	82	43	53

数组

- 数组元素的访问
  - 通过 **数组名 [下标]** 访问存储在数组中某一位置的值
    - 下标是指在元素在数组中的位置，从 0 开始
    - 下标的范围：$0 \sim L-1$ (L 为数组大小)
  - 例如：声明数组 `double b[5];`
    - 可以使用的数组下标：$0, 1, 2, 3, 4$
    - 对应的可以使用的数组元素：`b[0]`, `b[1]`, `b[2]`, `b[3]`, `b[4]`
    - 可以使用变量作为数组下标：`b[i]` ($0 \le i \le 4$)

`int a[8];`

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
71	80	62	91	99	82	43	53

例 9.1：成绩录入

编程题

- 6 年 A 班进行了一次小测，现要求编写程序，录入所有学生成绩，并输出所有学生的成绩，学生的成绩如下。

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
71	80	62	91	99	82	43	53

- 样例输入
  无
- 样例输出

```
71 80 62 91 99 82 43 53
```

例 9.1：成绩录入

```cpp
#include <iostream>

using namespace std;

// 录入成绩
int a[8] = {71, 80, 62, 91, 99, 82, 43, 53};

int main() {
  // 输出所有成绩
  cout <<
    a[0] << " " << a[1] << " " << a[2] << " " <<
    a[3] << " " << a[4] << " " << a[5] << " " <<
    a[6] << " " << a[7] << " " << endl;

  return 0;
}
```

例 9.2：成绩查询

编程题

- 已知 6 年 A 班的小测成绩如下，编写程序，输入一个整数 k $(0 \leq k < 8)$，输出 k 号学生的分数。

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
71	80	62	91	99	82	43	53

- 样例输入

  5

- 样例输出

  82

例 9.2：成绩查询

```cpp
#include <iostream>

using namespace std;

int a[8] = {71, 80, 62, 91, 99, 82, 43, 53};

int main() {
  int k;
  cin >> k;
  cout << a[k] << endl;

  return 0;
}
```

思考

当 $k = -1$ 时会有怎样的运行结果?

数组

- 数组访问越界
  - 一个大小为 L 的数组，那下标在 0 到 $L-1$ 的之间才是有意义的
  - 访问不在数组有意义区间的元素称为 **数组越界**

```
int a[8];
```

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
71	80	62	91	99	82	43	53

数组

- 数组访问越界
  - 一个大小为 L 的数组，那下标在 0 到 $L-1$ 的之间才是有意义的
  - 访问不在数组有意义区间的元素称为 **数组越界**
  - 这可能会导致程序出现重大错误，但编译不会报错，需要有意识地避免数组越界
  - 尤其是当数组的下标是以变量的形式出现时，更要加以小心

```
int a[8];
```

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
71	80	62	91	99	82	43	53

目录

数组元素的使用

- 数组元素的使用与一般变量的用法相同
- 例如：输入输出
  - `cin >> a[0] >> a[1] >> a[2];`
  - `cout << a[0] << " " << a[2] << endl;`

例 9.3：成绩修改 I

编程题

- 已知 6 年 A 班的小测成绩如下，其中，k 号同学发现自己试卷错判，实际分数应该高 x 分。
  编写程序，输入两个整数 k $(0 \le k < 8)$ 和 x $(0 \le x \le 100)$，请修改并输出 k 号同学的分数。

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
71	80	62	91	99	82	43	53

- 样例输入

```
2 6
```

- 样例输出

```
68
```

例 9.3：成绩修改 I

```cpp
#include <iostream>

using namespace std;

int a[8] = {71, 80, 62, 91, 99, 82, 43, 53};

int main() {
  int k, x;
  cin >> k >> x;
  a[k] += x; // 修改 k 号同学的成绩
  cout << a[k] << endl;

  return 0;
}
```

例 9.4：成绩修改 II

编程题

- 已知 6 年 A 班的小测成绩如下，其中，x 号与 y 号同学的成绩登记反了。
  编写程序，输入两个整数 x $(0 \leq x < 8)$ 和 y $(0 \leq y \leq 8)$，请交换他们的成绩，并输出正确的成绩表，以空格间隔。

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
71	80	62	91	99	82	43	53

- 样例输入

```
3 4
```

- 样例输出

```
71 80 52 99 91 82 43 53
```

例 9.4：成绩修改 II

```cpp
#include <iostream>

using namespace std;

int a[8] = {71, 80, 62, 91, 99, 82, 43, 53};

int main() {
  int x, y;
  cin >> x >> y;
  // 交换 x 与 y 的成绩
  int tmp = a[x];
  a[x] = a[y];
  a[y] = tmp;
  // 输出所有成绩
  cout <<
    a[0] << " " << a[1] << " " << a[2] << " " <<
    a[3] << " " << a[4] << " " << a[5] << " " <<
    a[6] << " " << a[7] << " " << endl;

  return 0;
}
```

例 9.5：计算书费 I

编程题

- 编写程序，输入两个整数 k $(0 \leq k < 5)$ 和 x $(0 \leq x \leq 100)$，表示购买编号为 k 的图书 x 本，输出消费的金额，结果保留两位小数，图书的单价如下。

b[0]	b[1]	b[2]	b[3]	b[4]
28.9	78.5	35.4	43.6	56

- 样例输入

```
3 5
```

- 样例输出

```
218.00
```

例 9.5：计算书费 I

```cpp
#include <iostream>
#include <iomanip>

using namespace std;

double b[] = {28.9, 78.5, 35.4, 43.6, 56}; // 录入单价

int main() {
  int k, x;
  cin >> k >> x;
  cout << fixed << setprecision(2) << b[k] * x << endl;

  return 0;
}
```

例 9.6：计算书费 II

编程题

- 编写程序，输入一个整数 n $(1 \leq n \leq 100)$，表示购买次数。然后输入 n 行，每行两个整数 k $(0 \leq k < 5)$ 和 x $(0 \leq x \leq 100)$，表示购买编号为 k 的图书 x 本，分别输出 n 次购书的消费金额，结果保留两位小数，图书的单价如下。

b[0]	b[1]	b[2]	b[3]	b[4]
28.9	78.5	35.4	43.6	56

- 样例输入

```
2
3 4
0 5
```

- 样例输出

```
174.40
144.50
```

例 9.6：计算书费 II

```cpp
#include <iostream>
#include <iomanip>

using namespace std;

double b[] = {28.9, 78.5, 35.4, 43.6, 56}; // 录入单价

int main() {
  int n;
  cin >> n;
  int k, x;
  for (int i = 0; i < n; i++) {
    cin >> k >> x;
    cout << fixed << setprecision(2) << b[k] * x << endl;
  }

  return 0;
}
```

思考

如何求 n 次购书的总额呢?

目录

数组的遍历

- 遍历：对数组中所有元素逐个访问一遍的过程称为遍历
- 遍历输入数组
  - 存储在下标 $0 \sim n-1$ 中
  - ```
    for (int i = 0; i < n; i++) {
      cin >> a[i];
    }
    ```
    ```

数组的遍历

- 遍历：对数组中所有元素逐个访问一遍的过程称为遍历
- 遍历输入数组
  - 存储在下标 $0 \sim n-1$ 中
  - ```
    for (int i = 0; i < n; i++) {
      cin >> a[i];
    }
    ```
  - 存储在下标 $1 \sim n$ 中
  - ```
    for (int i = 1; i <= n; i++) {
      cin >> a[i];
    }
    ```

数组的遍历

- 遍历：对数组中所有元素逐个访问一遍的过程称为遍历
- 遍历输出数组
  - 存储在下标 $0 \sim n-1$ 中
  - ```
    for (int i = 0; i < n; i++) {
      cout << a[i] << " ";
    }
    cout << endl;
    ```
    ```

数组的遍历

- 遍历：对数组中所有元素逐个访问一遍的过程称为遍历
- 遍历输出数组
  - 存储在下标 $0 \sim n-1$ 中
  - ```
    for (int i = 0; i < n; i++) {
      cout << a[i] << " ";
    }
    cout << endl;
    ```
  - 存储在下标 $1 \sim n$ 中
  - ```
    for (int i = 1; i <= n; i++) {
      cout << a[i] << " ";
    }
    cout << endl;
    ```

例 9.7：倒序输出数组元素

编程题

- 编写程序，输入一个整数 n $(1 \leq n \leq 100)$，表示有 n 个整数，接下来输入 n 个整数存储在数组中，要求倒序输出数组元素。
- 样例输入

  ```
  6
  1 4 2 8 5 7
  ```

- 样例输出

  ```
  7 5 8 2 4 1
  ```

例 9.7：倒序输出数组元素

```cpp
#include <iostream>

using namespace std;

int a[105];

int main() {
  int n;
  cin >> n;
  // 按照下标 1 ~ n 的顺序输入数组元素
  for (int i = 1; i <= n; i++) {
    cin >> a[i];
  }
  // 按照下标 n ~ 1 的顺序输出数组元素
  for (int i = n; i >= 1; i--) {
    cout << a[i] << " ";
  }
  cout << endl;

  return 0;
}
```

例 9.8：交替输出数组元素

编程题

- 编写程序，输入一个偶数 n ($1 \leq n \leq 100$)，接下来输入 n 个整数，然后交替输出。
  即按照：第 1 个数、倒数第 1 个数、第 2 个数、倒数第 2 个数 ... 的顺序输出，其中每行输出两个数字，数字与数字之间有空格隔开。

- 样例输入

```
6
1 4 2 8 5 7
```

- 样例输出

```
1 7
4 5
2 8
```

例 9.8：交替输出数组元素

编程题

- 编写程序，输入一个偶数 n $(1 \leq n \leq 100)$，接下来输入 n 个整数，然后交替输出。
  即按照：第 1 个数、倒数第 1 个数、第 2 个数、倒数第 2 个数 . . . 的顺序输出，其中每行输出两个数字，数字与数字之间有空格隔开。

- 样例输入

```
6
1 4 2 8 5 7
```

- 样例输出

```
1 7
4 5
2 8
```

例 9.8：交替输出数组元素

编程题

- 编写程序，输入一个偶数 n $(1 \leq n \leq 100)$，接下来输入 n 个整数，然后交替输出。
  即按照：第 1 个数、倒数第 1 个数、第 2 个数、倒数第 2 个数 . . . 的顺序输出，其中每行输出两个数字，数字与数字之间有空格隔开。

- 样例输入

```
6
1 4 2 8 5 7
```

- 样例输出

```
1 7
4 5
2 8
```

例 9.8：交替输出数组元素

编程题

- 编写程序，输入一个偶数 n $(1 \leq n \leq 100)$，接下来输入 n 个整数，然后交替输出。
  即按照：第 1 个数、倒数第 1 个数、第 2 个数、倒数第 2 个数 . . . 的顺序输出，其中每行输出两个数字，数字与数字之间有空格隔开。

- 样例输入

```
6
1 4 2 8 5 7
```

- 样例输出

```
1 7
4 5
2 8
```

例 9.8：交替输出数组元素

```cpp
#include <iostream>

using namespace std;

int a[105];

int main() {
  int n;
  cin >> n;
  // 按照下标 1 ~ n 的顺序输入数组元素
  for (int i = 1; i <= n; i++) {
    cin >> a[i];
  }
  // 交替输出数组元素
  for (int i = 1, j = n ; i <= j; i++, j--) {
    cout << a[i] << " " << a[j] << endl;
  }

  return 0;
}
```

思考

两个数组间可以像变量一样直接赋值吗?

数组间的赋值

```cpp
#include <iostream>

using namespace std;

int a[105], b[105];

int main() {
  int n;
  cin >> n;
  for (int i = 1; i <= n; i++) { // 输入数组 a
    cin >> a[i];
  }
  b = a;                         // 将数组 a 赋值给数组 b
  for (int i = 1; i <= n; i++) { // 输出数组 b
    cout << b[i] << " ";
  }
  cout << endl;

  return 0;
}
```

数组间的赋值

```cpp
#include <iostream>

using namespace std;

int a[105], b[105];

int main() {
  int n;
  cin >> n;
  for (int i = 1; i <= n; i++) { // 输入数组 a
    cin >> a[i];
  }
  b = a;                         // 将数组 a 赋值给数组 b
  for (int i = 1; i <= n; i++) { // 输出数组 b
    cout << b[i] << " ";
  }
  cout << endl;

  return 0;
}
```

- 编译会报错，数组之间不能直接赋值

数组间的赋值

- 数组之间的赋值不可以像变量那样直接进行赋值
  - `b = a;`

数组间的赋值

- 数组之间的赋值不可以像变量那样直接进行赋值
  - ~~`b = a;`~~
- 只能通过遍历依次进行赋值
  - ```
    for (int i = 1; i <= n; i++) {
      b[i] = a[i];
    }
    ```
    ```

数组间的赋值

```cpp
#include <iostream>

using namespace std;

int a[105], b[105];

int main() {
  int n;
  cin >> n;
  for (int i = 1; i <= n; i++) { // 输入数组 a
    cin >> a[i];
  }
  for (int i = 1; i <= n; i++) { // 将数组 a 赋值给数组 b
    b[i] = a[i];
  }
  for (int i = 1; i <= n; i++) { // 输出数组 b
    cout << b[i] << " ";
  }
  cout << endl;

  return 0;
}
```

目录

例 9.9：求数组元素的和

编程题

- 编写程序，输入一个整数 n $(1 \leq n \leq 100)$，表示有 n 个整数，接下来输入 n 个整数存储在数组中，求这 n 个数组元素之和。
- 样例输入

  ```
  6
  1 4 2 8 5 7
  ```

- 样例输出

  ```
  27
  ```

例 9.9：求数组元素的和

```cpp
#include <iostream>

using namespace std;

int a[105];

int main() {
  int n;
  cin >> n;
  for (int i = 1; i <= n; i++) {
    cin >> a[i];
  }
  int sum = 0;
  for (int i = 1; i <= n; i++) {
    sum += a[i];
  }
  cout << sum << endl;

  return 0;
}
```

例 9.10：求数组元素的最大值

编程题

- 编写程序，输入一个整数 n $(1 \leq n \leq 100)$，表示有 n 个整数，接下来输入 n 个整数存储在数组中，求这 n 个数组元素中的最大值。
- 样例输入

  ```
  6
  1 4 2 8 5 7
  ```

- 样例输出

  ```
  8
  ```

例 9.10：求数组元素的最大值

```cpp
#include <iostream>

using namespace std;

int a[105];

int main() {
  int n;
  cin >> n;
  for (int i = 1; i <= n; i++) {
    cin >> a[i];
  }
  int maxv = -1000000000;
  for (int i = 1; i <= n; i++) {
    if (a[i] > maxv) {
      maxv = a[i];
    }
  }
  cout << maxv << endl;

  return 0;
}
```

例 9.11：求数组元素的最大值及位置

编程题

- 编写程序，输入一个整数 n $(1 \leq n \leq 100)$，表示有 n 个整数，接下来输入 n 个整数存储在数组中，求这 n 个数组元素中的最大值及位置（保证最大值是唯一的），输出以空格间隔。
- 样例输入

  ```
  6
  1 4 2 8 5 7
  ```

- 样例输出

  ```
  8 4
  ```

例 9.11：求数组元素的最大值及位置

```cpp
#include <iostream>

using namespace std;

int a[105];

int main() {
  int n;
  cin >> n;
  for (int i = 1; i <= n; i++) {
    cin >> a[i];
  }
  int pos = 0;
  for (int i = 1; i <= n; i++) {
    if (pos == 0 || a[i] > a[pos]) {
      pos = i;
    }
  }
  cout << a[pos] << " " << pos << endl;

  return 0;
}
```

例 9.11：求数组元素的最大值及位置

```cpp
#include <iostream>

using namespace std;

int a[105];

int main() {
  int n;
  cin >> n;
  for (int i = 1; i <= n; i++) {
    cin >> a[i];
  }
  int pos = 0;
  for (int i = 1; i <= n; i++) {
    if (pos == 0 || a[i] > a[pos]) {
      pos = i;
    }
  }
  cout << a[pos] << " " << pos << endl;

  return 0;
}
```

目录

总结

- 数组的概念
- 数组的声明
- 数组的初始化
- 数组元素的访问
- 数组的应用
  - 数组求和
  - 求最值及其下标

Thank you!