西北工业大学
NORTHWESTERN POLYTECHNICAL UNIVERSITY

C++程序设计
Programming in C++

1011018

主讲：魏英，计算机学院

指针与数组

1、一维数组与指针

- C++程序员更偏爱使用指针来访问数组元素，这样做的好处是运行效率高、写法简洁。
-

- 1. 一维数组的地址
- 数组由若干个元素组成，每个元素都有相应的地址，通过取地址运算（&）可以得到每个元素的地址。

```
int a[10];
int *p=&a[0]; //定义指向一维数组元素的指针
p=&a[5]; //指向a[5]
```

- C++规定，数组名既代表数组本身，又代表整个数组的地址，还是是数组首元素的地址值，即a与第0个元素的地址&a[0]相同。例如下面两个语句是是等价的。

```
①p=a;
②p=&a[0];
```

- 数组名是一个指针常量，因而它不能出现在左值和某些算术运算中，例如：

```
int a[10], b[10], c[10];
a=b; //错误，a是常量不能出现在左值的位置
c=a+b; //错误，a、b是地址值，不允许加法运算
a++; //错误，a是常量不能使用++运算
```

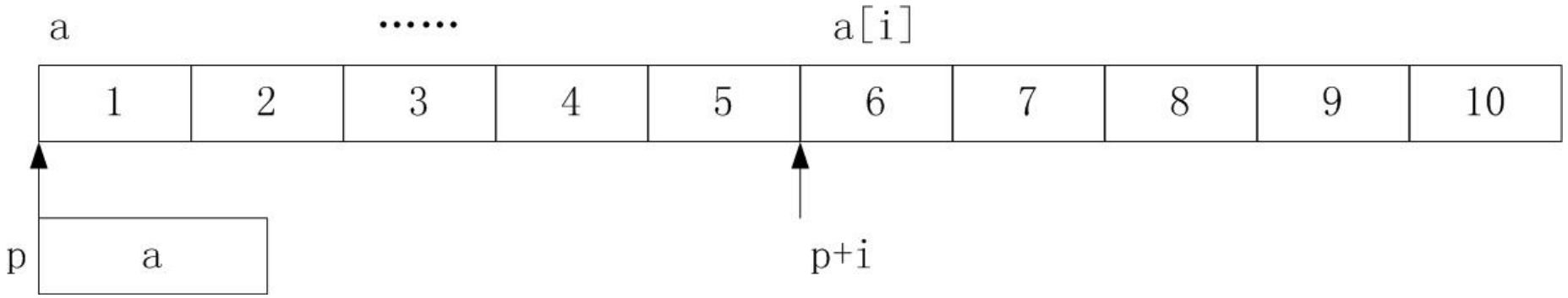
- 2. 指向一维数组的指针变量
- 定义指向一维数组元素的指针变量时，指向类型应该与数组元素类型一致，例如：

```
int a[10], *p1;
double f[10], *p2;
p1=a; //正确
p2=f; //正确
p1=f; //错误，指向类型不同不能赋值
```

- 3. 通过指针访问一维数组
- 由于数组元素的地址是规律性增加的，根据指针算术运算规则，可以利用指针及其运算来访问数组元素。
- 设有如下定义：

```
int *p, a[10]={1,2,3,4,5,6,7,8,9,10};
p=a; //p指向数组a
p++;
```

图18.1 指向一维数组的指针



设：a是一维数组名，p是指针变量且p=a。
根据以上叙述，访问一个数组元素a[i]，可以用：
①数组下标法：a[i]；
②指针下标法：p[i]；
③地址引用法：*(a+i)；
④指针引用法：*(p+i)。

【例6.1】用多种方法遍历一维数组元素

①下标法。

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      int a[10], i;
6      for (i=0;i<10;i++) cin>>a[i];
7      for (i=0;i<10;i++) cout<<a[i]<<" ";
8      return 0;
9  }
```

【例6.1】用多种方法遍历一维数组元素

②通过地址间接访问数组元素。

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      int a[10], i;
6      for (i=0;i<10;i++) cin>>*(a+i);
7      for (i=0;i<10;i++) cout<<*(a+i)<<" ";
8      return 0;
9  }
```

【例6.1】用多种方法遍历一维数组元素

③通过指向数组的指针变量间接访问元素。

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      int a[10], *p;
6      for (p=a;p<a+10;p++) cin>>*p;
7      for (p=a;p<a+10;p++) cout<<*p<<" ";
8      return 0;
9  }
```

- 4．数组元素访问方法的比较

- （1）使用下标法访问数组元素，程序写法比较直观，能直接知道访问的是第几个元素。
- （2）而使用指针引用法，指针变量直接指向元素，不必每次都重新计算地址，能提高运行效率。
- （3）将自增和自减运算用于指针变量十分有效，可以使指针变量自动向前或向后指向数组的下一个或前一个元素。

C程序设计