# 编程实现升序和降序排序

- 如果不使用函数指针编程…

```
//选择法升序排序
void AscendingSort(int a[], int n)
{
        int i, j, k;
        for (i=0; i<n-1; i++)
       {
          k = i;
          for (j=i+1; j<n; j++)
          {
              if (a[j] < a[k])  k = j;
          }
          if (k != i)
          {
              Swap(&a[k], &a[i]);
          }
        }
}
```

```
//选择法降序排序
void DescendingSort(int a[], int n)
{
        int i, j, k;
        for (i=0; i<n-1; i++)
       {
          k = i;
          for (j=i+1; j<n; j++)
          {
              if (a[j] > a[k])  k = j;
          }
          if (k != i)
          {
              Swap(&a[k], &a[i]);
          }
        }
}
```

# 编程实现升序和降序排序

- 使用函数指针编写一个通用的排序函数

```
//选择法升序排序
void SelectionSort(int a[], int n, int (*compare)(int, int))
{
        int i, j, k;
        for (i=0; i<n-1; i++)
        {
            k = i;
            for (j=i+1; j<n; j++)
            {
                if ((*compare)(a[j], a[k]))  k = j;
            }
            if (k != i)
            {
                Swap(&a[k], &a[i]);
            }
        }
}
```

增加一个函数指针形参

调用函数指针compare指向的函数

```
SelectionSort(score, n, Ascending);
SelectionSort(score, n, Descending);
```

# 编程实现升序和降序排序

- 使用函数指针编写一个通用的排序函数

```
//选择法升序排序
void SelectionSort(int a[], int n, int (*compare)(int, int))
{
        int i, j, k;
        for (i=0; i<n-1; i++)
        {
            k = i;
            for (j=i+1; j<n; j++)
            {
                if ((*compare)(a[j], a[k]))  k = j;
            }
            if (k != i)
            {
                Swap(&a[k], &a[i]);
            }
        }
}
```

```
int Ascending(int a, int b)
{
    return a < b; //为真，则升序
}
```

```
if (a[j] < a[k])
```

```
int Descending(int a, int b)
{
    return a > b; //为真，则降序
}
```

```
if (a[j] > a[k])
```

# 小结

- 正确理解指针的概念
  - 指针是一种特殊的数据类型
  - 指针类型的变量，称为指针变量
  - 指针不是地址，指针变量的值是一个地址
  - 想让指针变量指向哪个存储单元，就让其保存哪个单元的地址
    - 保存一个变量的地址
    - 保存一个数组的首地址
    - 保存一个字符串的首地址
    - 保存一个函数的入口地址

# 小结

- 使用指针变量的基本原则
  - 明确指针指向了哪里——初始化的目的
  - 明确指针指向单元的内容是什么——基类型
  - 只能指向同一基类型的数据——一个（x型）的指针指向一个（x型）的变量

# 小结

- 指针的重要应用
  - 作函数参数，向函数传递变量或函数的地址
  - 动态分配内存，实现动态数组和动态数据结构
- 指向变量的指针作函数参数
  - 被调函数根据传入的地址读写它不能直接访问的变量的值
- 指向函数的指针，作函数参数
  - 被调函数根据传入的不同地址调用不同的函数