BÀI 1: SẮP XẾP

1. **Chuẩn đầu ra:** Sau bài này, người học có thể
   - ✓ Xây dựng các chương trình thực hiện sắp xếp.
   - ✓ Áp dụng giải thuật sắp xếp cho các kiểu dữ liệu thực tế
2. **Chuẩn bị:**
   - ✓ Đọc trước lý thuyết các giải thuật sắp xếp
3. **Phương tiện:**
   - ✓ Sử dụng 1 trong các phần mềm sau: Borland C 3.1, Visual Studio 6.0, Visual Studio .NET.
4. **Thời lượng: 5 tiết**
5. **Nội dung thực hành:**

Cài đặt các hàm sắp xếp:

1. **SelectionSort**
2. **InsertionSort**
3. **BubbleSort**
4. **QuickSort**

Phần nâng cao:

1. **InsertionSort cải tiến (sử dụng tìm kiếm nhị phân)**
2. **ShakerSort**

Áp dụng cho **kiểu số nguyên** và **kiểu SINHVIEN** (sắp xếp theo mã số sinh viên, điểm trung bình).

Kiểu SINHVIEN được định nghĩa như sau:

```
typedef struct
{
   int mssv;
   char hoten[50];
```

```
    float dtb;
}SINHVIEN;
```

Lưu ý:

+ Viết hàm main cho phép người dùng lựa chọn 1 trong số các phương pháp sắp xếp.

+ Xây dựng theo kiểu dữ liệu chung ElementType như sau:

#define ElementType int (SINHVIEN, ….).

Khi đó các **toán tử so sánh** cần phải được viết riêng cho từng kiểu dữ liệu:

```
//Kiểu số nguyên
int SoSanh(int a, int b)
{
    if(a>b)
        return 1;
    return 0;
}
//Kiểu SINHVIEN (so sánh theo mã số sinh viên)
int SoSanh(SINHVIEN a, SINHVIEN b)
{
    if(a.mssv>b.mssv)
        return 1;
    return 0;
}
```

Hàm sắp xếp được viết theo kiểu dưới đây.

```
void SelectionSort(ElementType A[], int n)
{
    int minPos;
    for(int i=0;i<n-1;i++)
    {
        //Tìm vị trí phần tử nhỏ nhất
        minPos=i;
```

```
        for(int j=i+1;j<n;j++)
                        if(SoSanh(A[j],A[minPos]))
                                minPos=j;
        //Hoán vị A[i], A[minIndex]
        if(minPos!=i)
        {
                ElementType temp=a[i];
                a[i]=a[minPos];
                a[minPos]=temp;
        }
    }
}
```

//Có thể thấy là hàm SelectionSort phía trên có thể chạy cho cả kiểu dữ liệu số nguyên hoặc SinhVien.