BÁO CÁO THỰC HÀNH

Họ tên	Nguyễn Việt Nhật	Lớp: KHTN2021
MSSV	21520378	STT: 15
Bài Thực Hành	LAB05	
CBHD	**Trương Văn Cương**	

Điểm buổi thực hành

Chuyên cần (20%)		
Trình bày (20%)		
Nội dung thực hành (60%)		
Tổng (100%)		

Bài tập thực hành:

Viết chương trình Assembly thực hiện các yêu cầu sau:

1. Nhập một mảng số tự nhiên gồm ***N*** phần tử (*0 < N < 101*)
2. Sắp xếp các phần tử trong dãy tăng dần theo các thuật toán sau:
    - Bubble sort
    - Selection sort
    - Insertion sort

1 Giới thiệu về các thuật toán:

1.1 Sắp xếp nổi bọt (Bubble sort)

Đây là thuật toán cơ bản nhất cho việc sắp xếp.

Ý tưởng:

- Xét lần lượt các cặp 2 phần tử liên tiếp. Nếu phần tử đứng sau nhỏ hơn phần tử đứng trước, ta đổi chỗ 2 phần tử.
- Lặp lại đến khi không còn 2 phần tử nào thỏa mãn.

1.2 Sắp xếp chèn (Insertion sort)

Ý tưởng:

- Sắp xếp lần lượt từng đoạn gồm 1 phần tử đầu tiên, 2 phần tử đầu tiên, ... cho đến ***N*** phần tử.
- Với mỗi đoạn ***i*** phần tử đã được sắp xếp, ta có thể sắp xếp ***i + 1*** phần tử đầu tiên bằng cách tìm vị trí phù hợp của phần tử thứ ***i + 1*** và "chèn" nó vào đó.

1.3 Sắp xếp chọn (Selection sort)

Ý tưởng:

- Tìm phần tử có giá trị nhỏ nhất trong đoạn chưa được sắp xếp và đổi chỗ phần tử nhỏ nhất đó với phần tử ở đầu đoạn chưa được sắp xếp *(không phải đầu mảng)*.
- Thuật toán sẽ chia mảng thành 2 mảng con:
  - Một mảng con đã được sắp xếp *(bên trái)*.
  - Một mảng con chưa được sắp xếp *(bên phải)*.
- Tại mỗi bước lặp của thuật toán, phần tử nhỏ nhất ở mảng con chưa được sắp xếp sẽ được di chuyển về đoạn đã sắp xếp.

2 Minh họa

Truy cập vào trang web VisuAlgo

- Chọn **Tên thuật toán** ở menu phía bên trên.
- Chọn **Create** để tạo mảng mới theo ý muốn.
- Bấm **Sort** để xem minh họa thuật toán.

3 Pseudocode

3.1 Nhập mảng gồm N số tự nhiên (Input)

```
get number of elements (N)   // $s0 <- N
while (N < 0 or N > 100)     // break when 0 < $s0 < 101
  get number of elements
end

// read elements
set index to 0         // $t1 = 0
while (index < N)                    // $t1 < $s0
  get value of i-th element of array // first: $t8 <- value - then: arr[index] <- $t8
  while (array[index] <= 0)
        get value of i-th element of array
  end
end
```

3.2 Sắp xếp nổi bọt (Bubble sort)

```
for i = 0 to N - 1 do:
  swapped = false

  for j = 0 to N - 1 do :

        /* compare the adjacent elements */
        if array[j] > array[j + 1] then
              /* swap them */
              swap(array[j], array[j + 1])
              swapped = true
        end if
  end for

  /*if no number was swapped that means
  array is sorted now, break the loop.*/

  if (not swapped) then
        break
  end if
end for
```

3.3 Sắp xếp chèn (Insertion sort)

```
int holePosition
int valueToInsert

for i = 1 to N - 1 do:

  /* select value to be inserted */
  valueToInsert = A[i]
  holePosition = i

  /*locate hole position for the element to be inserted */

  while holePosition > 0 and A[holePosition - 1] > valueToInsert do:
         A[holePosition] = A[holePosition - 1]
         holePosition = holePosition - 1
  end while

  /* insert the number at hole position */
  A[holePosition] = valueToInsert
 end for
```

3.4 Sắp xếp chọn (Selection sort)

```
for i = 0 to N - 1
  /* set current element as minimum*/
  mini = i

  /* check the element to be minimum */

  for j = i + 1 to N - 1
         if array[j] < array[mini] then
                mini = j;
         end if
  end for

  /* swap the minimum element with the current element*/
  if indexMin != i  then
         swap array[min] and array[i]
  end if
 end for
```

3.5 Xuất mảng đã sắp xếp

```
for i = 0 to N - 1
  print array[i]
```

4 Thông tin chi tiết

4.1 Repo Github

4.2 Source code

4.3 Report

5 Nguồn tham khảo

- Topcoder
- VisuAlgo
- Wikipedia