## Bài tập số 05

(Deadline: 02 tuần kể từ ngày giao – Ngày giao: Ngày 07 tháng 5 năm 2016)

#### Mục tiêu:

- Con trỏ hàm, Con trỏ trỏ con trỏ, Con trỏ kép và mảng 2 chiều
- Làm việc với bộ nhớ động
- Rèn luyện kỹ năng áp dụng các kiến thức trên vào xây dựng chương trình giải quyết bài toán thực tế
- Chú ý 1: Với các yêu cầu, cần biết cách thiết kế và xây dựng hàm, cách sử dụng hàm đã xây dựng được
- Chú ý 2: SV là chữ số cuối cùng của thẻ sinh viên
- Chú ý 3: Tên file mã nguồn chương trình vẫn theo quy định trong hướng dẫn. Tuy nhiên, các bài được tạo thành 1 file hoàn chỉnh, các phần chỉ là các hàm con trong đó
- Chú ý 4: Phần đầu mã mỗi chương trình, sử dụng cú pháp chú thích nhiều dòng /\* \*/ thêm vào thông tin Họ và tên (tiếng Việt không dấu), Mã lớp, mã thẻ sinh viên

### Phần I: Con trỏ hàm

**5.01**. Thực hiện xây dựng các hàm thực hiện: Hàm đọc từ một file dữ liệu ra thông số chiều cao N của các hình sau (chiều cao của hình được định nghĩa là số hàng mà hình đó sẽ chiếm) − file dạng text, trong file chứa duy nhất một số nguyên N ở hàng đầu tiên, file kết thúc bởi ký hiệu EOF; Hàm thực hiện vẽ các hình sau bằng cách sử dụng các cấu trúc điều kiển (lặp, lựa chọn, ...), mỗi hình tương ứng 1 hàm, trong đó ngoài tham số là chiều cao N của hình cần vẽ (đọc ra từ file dữ liệu), hàm còn có thêm một tham số là con trỏ file để lựa chọn in (vẽ) hình ra màn hình hay in (vẽ) vào file kết quả. Một mảng con trỏ hàm với số phần tử bằng số hàm vẽ (bằng số hình cần vẽ, vì mỗi hình vẽ cần một hàm). Sử dụng mảng con trỏ hàm (array of pointers to function) này trong hàm main() − Coi mỗi ô là khoảng độ rộng một ký tự, ô trống tương ứng ký tự trắng (**SV mod 4 ≡ 0(a,e,j); 1(b,g,k); 2(c,f,i); 3(h,I,m)**)

| *        | *                   | *                 | * | * | * | * | * | * |  | \             | * | * | *      | *         | * | *      | *           | /    |                | *              | * | * | * | * | * | * | * | * |  |  |  |
|----------|---------------------|-------------------|---|---|---|---|---|---|--|---------------|---|---|--------|-----------|---|--------|-------------|------|----------------|----------------|---|---|---|---|---|---|---|---|--|--|--|
| *        | *                   | *                 | * |   | * | * | * | * |  | *             | \ | * | *      | *         | * | *      | /           | *    |                |                | * | * | * | * | * | * | * | 1 |  |  |  |
| *        | *                   | *                 |   |   |   | * | * | * |  | *             | * | \ | *      | *         | * | /      | *           | *    |                |                |   | * | * | * | * | * |   |   |  |  |  |
| *        | *                   |                   |   |   |   |   | * | * |  | *             | * | * | \      | *         | / | *      | *           | *    |                |                |   |   | * | * | * |   |   |   |  |  |  |
| *        |                     |                   |   |   |   |   |   | * |  | *             | * | * | *      | Х         | * | *      | *           | *    |                |                |   |   |   | * |   |   |   |   |  |  |  |
| *        | *                   |                   |   |   |   |   | * | * |  | *             | * | * | /      | *         | \ | *      | *           | *    |                |                |   |   |   |   |   |   |   |   |  |  |  |
| *        | *                   | *                 |   |   |   | * | * | * |  | *             | * | / | *      | *         | * | \      | *           | *    |                |                |   |   |   |   |   |   |   |   |  |  |  |
| *        | *                   | *                 | * |   | * | * | * | * |  | *             | / | * | *      | *         | * | *      | \           | *    | _              |                |   |   |   |   |   |   |   |   |  |  |  |
| *        | *                   | *                 | * | * | * | * | * | * |  | /             | * | * | *      | *         | * | *      | *           | \    | (-) \          |                |   |   |   |   |   |   |   |   |  |  |  |
|          | (a) N=9, lẻ         |                   |   |   |   |   |   |   |  | (b) N=9, lẻ   |   |   |        |           |   |        |             |      |                | (c)N= 5, lẻ    |   |   |   |   |   |   |   |   |  |  |  |
|          |                     |                   |   | * |   |   |   |   |  |               |   |   |        | *         |   |        |             |      |                |                |   |   |   | 1 |   |   |   |   |  |  |  |
|          |                     |                   | * |   | * |   |   |   |  |               |   |   | *      | *         | * |        |             |      |                |                |   |   | 1 | 2 | 1 |   |   |   |  |  |  |
|          |                     | *                 |   | * |   | * |   |   |  |               |   | * | *      | *         | * | *      |             |      |                |                |   | 1 | 2 | 3 | 2 | 1 |   |   |  |  |  |
|          | *                   |                   | * |   | * |   | * |   |  |               | * | * | *      | *         | * | *      | *           |      |                |                | 1 | 2 | 3 | 4 | 3 | 2 | 1 |   |  |  |  |
| *        | * * * * *           |                   |   |   |   |   |   |   |  | *             | * | * | *      | *         | * | *      | *           | *    |                | 1              | 2 | 3 | 4 | 5 | 4 | 3 | 2 | 1 |  |  |  |
|          |                     |                   |   |   |   |   |   |   |  |               | * | * | *      | *         | * | *      | *           |      |                |                |   |   |   |   |   |   |   |   |  |  |  |
|          |                     |                   |   |   |   |   |   |   |  |               |   | * | *      | *         | * | *      |             |      |                |                |   |   |   |   |   |   |   |   |  |  |  |
|          |                     |                   |   |   |   |   |   |   |  |               |   |   | *      | *         | * |        |             |      |                |                |   |   |   |   |   |   |   |   |  |  |  |
|          | (e) N=5             |                   |   |   |   |   |   |   |  |               |   |   |        |           |   |        |             |      |                | (g) N=5 (N<=9) |   |   |   |   |   |   |   |   |  |  |  |
| <b>—</b> |                     | _                 | _ |   |   | _ | _ |   |  | (f) N=9, lẻ   |   |   |        |           |   |        |             |      | (g) N-3 (N<-9) |                |   |   |   |   |   |   |   |   |  |  |  |
| A        | В                   | С                 | D | Е | D | С | В | Α |  |               |   |   | 2      | 1         | _ |        |             |      |                | *              | * |   |   |   |   |   |   |   |  |  |  |
| A        | В                   | С                 | D |   | D | C | В | Α |  |               |   | 2 | 2      | 3         | 2 | 2      |             |      |                | *              | * | * |   |   |   |   |   |   |  |  |  |
| A        | В                   | С                 |   |   |   | C | В | Α |  |               | 4 | 3 | 4      | 5<br>7    | 4 | 3      | 4           |      |                | *              | * | * | * |   |   |   |   |   |  |  |  |
| A        | В                   |                   |   |   |   |   | В | Α |  | _             | 4 | 5 | 6      |           | 6 | 5<br>7 | 4           | Г    |                | *              | * | * | * | * |   |   |   |   |  |  |  |
| LA       | (b) N 5 (N 4 25) 12 |                   |   |   |   |   |   |   |  | 5             | 6 | 7 | (i) NI | 9<br>-E/N | 8 |        | 6           | 5    |                | *              | * | * | * | , |   |   |   |   |  |  |  |
|          | (h) N=5 (N<=25), lẻ |                   |   |   |   |   |   |   |  | (i) N=5(N<=5) |   |   |        |           |   |        |             |      |                | *              | * | * |   |   |   |   |   |   |  |  |  |
|          |                     |                   |   |   |   |   |   |   |  |               |   |   | *      | *         |   |        |             |      |                |                |   |   |   |   |   |   |   |   |  |  |  |
|          |                     |                   |   |   |   |   |   |   |  |               |   |   | *      |           |   |        |             |      |                |                |   |   |   |   |   |   |   |   |  |  |  |
|          |                     |                   |   |   |   |   |   |   |  |               |   |   |        |           |   | /i\    | N=9         | امًا |                |                |   |   |   |   |   |   |   |   |  |  |  |
| 6        | 5                   | 5 4 3 2 1 0 1 1 1 |   |   |   |   |   |   |  |               |   |   |        | *         | * | *      | (J <i>)</i> | *    | , IE<br>*      | *              | * | * |   |   |   |   |   |   |  |  |  |
| 0        | Э                   | 4                 |   | 2 | 1 | U |   |   |  |               |   |   |        | 1         | l |        |             |      |                |                |   |   |   |   |   |   |   | - |  |  |  |

| Г              | 4 | 3 | 2 | 1 | 0 |  |                     |   |   |   | 1 | 2 |  |  |  |  | Т           |   | * | * | * | * | * | * | * |   |
|----------------|---|---|---|---|---|--|---------------------|---|---|---|---|---|--|--|--|--|-------------|---|---|---|---|---|---|---|---|---|
| 5              | 4 | Э |   |   | U |  |                     |   |   |   | Т |   |  |  |  |  |             |   |   |   |   |   |   |   |   |   |
| 4              | 3 | 2 | 1 | 0 |   |  |                     |   |   | 1 | 2 | 3 |  |  |  |  |             |   |   | * | * | * | * | * |   |   |
| 3              | 2 | 1 | 0 |   |   |  |                     |   | 1 | 2 | 3 | 4 |  |  |  |  |             |   |   |   | * | * | * |   |   |   |
| 2              | 1 | 0 |   |   |   |  |                     | 1 | 2 | 3 | 4 | 5 |  |  |  |  |             |   |   |   |   | * |   |   |   |   |
| 1              | 0 |   |   |   |   |  |                     |   | 1 | 2 | 3 | 4 |  |  |  |  |             |   |   |   | * | * | * |   |   |   |
| 0              |   |   |   |   |   |  |                     |   |   | 1 | 2 | 3 |  |  |  |  |             |   |   | * | * | * | * | * |   |   |
|                |   |   |   |   |   |  |                     |   |   |   | 1 | 2 |  |  |  |  |             |   | * | * | * | * | * | * | * |   |
|                |   |   |   |   |   |  |                     |   |   |   |   | 1 |  |  |  |  |             | * | * | * | * | * | * | * | * | * |
| (k) N=7(N<=10) |   |   |   |   |   |  | (I) N=9 (N<=17), lẻ |   |   |   |   |   |  |  |  |  | (m) N=9, lẻ |   |   |   |   |   |   |   |   |   |

- **5.02**. Trong hàm sắp xếp, chúng ta đôi khi cần sắp xếp tăng dần, đôi khi cần sắp xếp giảm dần. Sự khác biệt cơ bản nằm ở chỗ là áp dụng việc so sánh > hay <. Trong phần này, chúng ta sẽ áp dụng con trỏ hàm vào phần so sánh của bài toán sắp xếp nhằm giảm nhỏ phần code chương trình. Để làm điều đó, thực hiện các công việc sau:
- a) Xây dựng một hàm làm nhiệm vụ hoán đổi giá trị của hai biến tên là hoandoi(): hàm có nhiệm vụ thực hiện việc hoán đổi giá trị của hai biến sau khi chúng được so sánh và cần hoán đổi. Ví dụ trong sắp xếp giảm dần của một mảng, giả sử chúng ta có a[1] = 8, a[3] = 12; khi đó giá trị 8 và 12 cần phải hoán đổi để 12 đứng trước 8, và hàm hoán đổi làm việc này. Hàm này không trả về giá trị.
- b) Xây dựng hai hàm so sánh sosanhhon() và sosanhkem(): hàm có nhiệm vụ so sánh giá trị của hai biến số và trả lại kết quả so sánh là đúng hay sai. Ví dụ: chúng ta cần so sánh hơn của a[1]=8 với a[3]=12, hàm sosanhhon(a[1],a[3]) sẽ cho kết quả false (0), trong khi hàm sosanhkem(a[1],a[3]) sẽ cho kết quả true (1)
- c) Xây dựng một con trỏ hàm để trỏ đến một trong hai hàm xây dựng ở câu b)
- d) Sử dụng con trỏ hàm ở câu c) để xây dựng hàm sắp xếp theo thuật toán "nổi bọt" (buble sort): hàm cho phép chỉ với một thực thi hàm (implementation), nhưng sử dụng con trỏ hàm sẽ cho phép sắp xếp tăng dần, hoặc giảm dần
- e) Kiểm tra hoạt động của hàm xây dựng được trong câu d) trong hàm main() với một dãy số được tạo ra ngẫu nhiên

### Phần II: Làm việc với bộ nhớ động (Dynamic memory allocation)

**5.03**. Cho một biến con trỏ tên là xPtr trỏ đến một giá trị kiểu nguyên. Không dùng biến trung gian, hãy tìm cách đọc trực tiếp từ bàn phím vào một giá trị nguyên cho vùng dữ liệu mà con trỏ này trỏ đến. Việc dùng biến trung gian như phần dưới đây không được chấp nhận

```
int *xPtr
int x;
xPtr = &x;
scanf("%d",&x);
```

- **5.04**. Để hạn chế việc sử dụng bộ nhớ quá lớn cho việc khai báo mảng một cách không cần thiết dẫn đến lãng phí vùng nhớ chương trình, bộ nhớ động sẽ cho phép tạo ra các vùng nhớ vừa (phù hợp) với yêu cầu của mảng mà người sử dụng cần. Gọi một biến con trỏ tên là xPtr là biến trỏ đến một vùng nhớ ứng với mảng mà người sử dụng yêu cầu. Hãy xây dựng chương trình gồm những hàm thực hiện các công việc sau :
- a) Hàm thực hiện việc cấp phát một vùng nhớ có kích thước N (là kích thước mảng mà người sử dụng mong muốn) cho con trỏ xPtr
- b) Hàm thực hiện yêu cầu người sử dụng nhập từ bàn phím vào các giá trị của mảng gồm N phần tử mà họ yêu cầu, trong đó mảng được tạo ra trong câu a)
- c) Hàm thực hiện tạo ra các giá trị ngẫu nhiên nằm trong khoảng mong muốn nào đó cho mảng gồm N phần tử mà người sử dụng yêu cầu, trong đó mảng được tạo ra trong câu a)
- d) Hàm thực hiện đọc từ file (dạng text: số đầu tiên là số phần tử của mảng, các số sau đó là các giá trị của phần tử trong mảng, mỗi số trên một hàng): tạo mảng có số phần tử mong muốn (số đầu tiên), gán các giá trị cho các phần tử của mảng
- e) Hàm in các giá trị của mảng tạo ra ở trên (trong các câu b), c), và d))
- f) Hàm tìm phần tử lớn nhất; hàm tìm phần tử nhỏ nhất; hàm đếm số phần tử chẵn trong mảng của các mảng có được trong các câu b), c) và d)
- g) Thử nghiệm các hàm xây dựng được ở trên trong hàm main()
- 5.05. Xây dựng chương trình gồm những hàm sau:
- a) Hàm cấp phát một vùng nhớ có kích thước N (là kích thước mảng mong muốn)

- b) Hàm tạo các giá trị ngẫu nhiên trong một khoảng xác định nào đó cho mảng được tạo ra trong câu a)
- c) Hàm kiểm tra, nếu kích thước N của mảng trong câu a) không phải là số dạng  $2^k$  với k là một số nguyên dương nào đó thì thực hiện mở rộng mảng đó ra thành mảng có kích thước dạng  $2^k$  với giá trị  $2^k$  lớn hơn kích thước cũ và gần kích thước cũ nhất (ví dụ: kích thước cũ là 6, dạng  $2^k$  lớn hơn 6 là 8, 16, ...; nhưng gần với 6 nhất là 8); các phần tử thêm vào được khởi tạo giá trị bằng 0
- d) Hàm in các kết quả mảng trong câu b), c) ra màn hình ; Hàm in các kết quả mảng trong câu b), c) ra file
- e) Trong hàm main(), yêu cầu người sử dụng nhập vào kích thước N của mảng mong muốn, thử nghiệm các hàm xây dựng được ở trên

# Phần III: Con trỏ trỏ con trỏ (Pointer to pointer)

5.06. Xét một đoạn code chương trình sau

```
int i=5, j=6, k=7;
int *ptr1, *ptr2;
int **ptr2ptr;
```

Hãy viết chương trình sử dụng con trỏ ptr2ptr thực hiện: tăng các giá trị của i, j, k lên 1 đơn vị; giảm các giá trị của i, j, k đi 2 đơn vị; tăng các giá trị của i, j, k gấp đôi

# Phần IV: Con trỏ trỏ con trỏ và mảng nhiều chiều

- **5.07**. Mặc dù không có sự tương đương giữa con trỏ trỏ con trỏ (còn gọi là con trỏ kép) và mảng 2 chiều như chúng ta thấy giữa con trỏ và mảng một chiều trong phần trước, nhưng chúng ta hoàn toán có thể sử dụng con trỏ kép để giả lập mảng 2 chiều khi kết hợp với việc cấp phát bộ nhớ động. Hãy viết một chương trình sử dụng con trỏ kép, kết hợp với cấp phát bộ nhớ động:
- Tạo ra một vùng nhớ tương ứng với một mảng 2 chiều có kích thước MxN (M, N do người sử dụng nhập vào từ bàn phím)
- Yêu cầu người sử dụng nhập vào các giá trị của mảng 2 chiều có kích thước MxN và chứa vào vùng nhớ tạo ra ở trên
- Thực hiện tìm phần tử lớn nhất, nhỏ nhất của mảng; phần tử lớn nhất, nhỏ nhất của mảng theo hàng; phần tử lớn nhất, nhỏ nhất của mảng theo cột; tổng giá trị các phần tử của mảng; tích giá trị các phần tử của mảng
- Tạo ngẫu nhiên các giá trị cho các phần tử mảng với giá trị ngẫu nhiên nằm trong một khoảng nào đó; sau đó lặp lại phần gạch đầu dòng thứ 3
- **5.08**. Lặp lại câu 5.07, nhưng các ý được đóng gói thành các hàm con, trong đó ngoài tham số chính là con trỏ kép còn có các tham số cần thiết khác (nếu cần)
- **5.09**. Sử dụng con trỏ (hoặc / và con trỏ kép), cùng với việc cấp phát bộ nhớ động và các ý tưởng trong câu 5.07 và các ý tưởng trong phần II, thực hiện các công việc sau với các mảng:
- a) Xây dựng một hàm với mảng 1D đã cho có kích thước N, trả về cho một mảng có các phần tử ở các vị trí cách nhau L của mảng đã cho

```
Ví dụ: a=\{0,1,2,3,4,5\}, L=1 \Rightarrow a' = \{0,2,4\}
```

b) Xây dựng một hàm với mảng 1D đã cho có kích thước N, trả về một mảng có kích thước xác định thích hợp trong đó giữa các phần tử của mảng cũ có L giá trị 0 được chèn vào

```
Ví dụ: a=\{-1,1,2,3,4,5\}, L=1 \Rightarrow a' = \{-1,0,1,0,2,0,3,0,4,0,5\}
```

c) Xây dựng một hàm với mảng 1D đã cho kích thước N, với hai tham số m, n, trả lại một mảng 2 chiều (giả lập bằng con trỏ kép) có kích thước mxn, trong đó các hàng của ma trận kết quả lần lượt là các phần tử của mảng ban đầu; nếu mxn nhỏ hơn N, hàm trả về giá trị -1, số phần tử thừa bị bỏ đi; nếu mxn bằng N, hàm trả về giá trị 0; nếu mxn lớn hơn N, hàm trả về giá trị 1, các giá trị thiếu được khởi đầu bằng giá trị 0

```
Ví dụ: a = {0, 1, 2, 3, 4, 5}, m=2, n=3 => A ={{0,1,2},{3,4,5}}, hàm trả về 0
```

**5.10**. Sử dụng con trỏ (hoặc / và con trỏ kép), cùng với việc cấp phát bộ nhớ động và các ý tưởng trong câu 5.07 và các ý tưởng trong phần II, thực hiện các công việc sau với các mảng:

Xây dựng hàm với 3 tham số cần thiết: nStart, nStep, nStop. Hàm trả lại một mảng 1 chiều có giá trị là một dãy số bắt đầu từ nStart đến nStop, mỗi giá trị trong dãy khác nhau nStep

```
Ví dụ: nStart = 0; nStep = 1; nStop = 5 => a = {0,1,2,3,4,5}
```