# BÀI TẬP VÀ THỰC HÀNH 3

## A. Mục đích, yêu cầu

- Củng cố cho học sinh những hiểu biết về kiểu dữ liệu mảng;
- Tổ chức những hoạt động trong phòng máy để học sinh có được các kĩ năng cơ bản làm việc với kiểu mảng (một chiều) trong lập trình, cụ thể là:

– Khai báo kiểu dữ liệu mảng;

– Nhập dữ liệu cho mảng, đưa ra màn hình chỉ số và giá trị các phần tử của mảng;

– Duyệt qua tất cả các phần tử của mảng để xử lí từng phần tử.

- Cung cấp cho học sinh ba thuật toán cơ bản và đơn giản thường gặp với dữ liệu kiểu mảng:

– Tính tổng các phần tử thoả mãn điều kiện nào đó;

– Đếm số các phần tử thoả mãn điều kiện nào đó;

– Tìm phần tử lớn nhất (hay nhỏ nhất) của mảng cùng với vị trí của nó trong mảng.

- Giới thiệu hàm *random*(*N*) cho học sinh thấy có thể dùng lệnh để máy lấy ngẫu nhiên một số nguyên trong khoảng từ 0 đến *N*−1, giới hạn *N* do người lập trình đưa ra.
- Góp phần hình thành và rèn luyện tư duy lập trình, tác phong của người lập trình.

## B. Những điểm cần lưu ý và gợi ý tổ chức thực hành

Khi kết thúc giờ thực hành, ít nhất học sinh phải chạy được ba chương trình ở bài 1*a*, 1*b* và bài 2*a*. Tuỳ theo điều kiện cụ thể, trình độ học sinh, giáo viên cần giúp học sinh đạt được mục đích chính của bài thực hành. Giáo viên có thể chuẩn bị sẵn chương trình ở câu *a* bài 1 trên đĩa (đĩa cứng của từng máy hay đĩa mềm) và cung cấp cho một số học sinh vào những thời điểm thích hợp.

***Về bài 1***

- Học sinh phải hiểu được chương trình cho sẵn ở câu *a* và thấy được kết quả chạy chương trình này, trên cơ sở đó mới tìm ra được cách giải quyết yêu cầu đặt ra ở câu *b*.
- Cần làm cho học sinh thấy được điểm khác nhau cơ bản của các chương trình ở câu *a* và câu *b* là chương trình ở câu *a* kiểm tra (lần lượt) từng phần tử của mảng để quyết định có cộng tích luỹ hay không, còn chương trình ở câu *b* kiểm tra từng phần tử của mảng để quyết định đếm tăng cho số các số dương hay đếm tăng cho số các số âm.
- Thủ tục *randomize* khởi tạo bộ lấy số ngẫu nhiên trước khi dùng lệnh *random*(*n*) để lấy một số nguyên ngẫu nhiên trong phạm vi từ 0 đến *n*−1.
- Nên ra một số câu hỏi phù hợp với trình độ của học sinh mỗi lớp, hướng dẫn học sinh tìm hiểu chương trình. Sau đây là một vài câu hỏi làm ví dụ:

– *MyArray* là tên một kiểu dữ liệu hay tên một biến?

– Vai trò của *nmax* và *n* khác nhau như thế nào? Có thể bỏ khai báo hằng *nmax* và viết *n* thay cho *nmax* trong khai báo kiểu mảng được không? Vì sao?

– Những dòng lệnh nào nhằm tạo ra mảng *A*?

– Lệnh gán:

```
A[i]:= random(301)-random(301);
```

có ý nghĩa gì?

– Câu lệnh *for-do* cuối cùng thực hiện nhiệm vụ nào trong những nhiệm vụ đặt ra cho chương trình? Giải thích?

– Câu lệnh:

```
s:= s+A[i];
```

được thực hiện bao nhiêu lần?

(bao nhiêu là tuỳ theo mảng *A* có bao nhiêu phần tử thoả mãn điều kiện là bội của số cho trước, số này được lưu trữ bởi biến *k*).

– Cho biết ý nghĩa của biến *posi*, cho biết ý nghĩa của biến *neg*. Vì sao lại biết được biến *posi* dùng để đếm số lượng phần tử có giá trị dương của mảng?

- Nếu đủ thời gian cũng có thể nêu thêm một vài yêu cầu khác để tăng thêm khả năng vận dụng của học sinh. Chẳng hạn, sửa lại chương trình để có thể nhập dữ liệu từ bàn phím. Tuy nhiên, do không có thời gian để nhập nhiều dữ liệu nên khi chạy thử chương trình hãy sửa kích thước tối đa của mảng là một số không vượt quá 10.
- Chú ý nhắc học sinh ghi lại chương trình ở câu *a* trước khi sửa đổi để có chương trình đáp ứng yêu cầu ở câu *b*. Nên dùng **Save as...** để lưu chương trình đã chạy tốt ở câu *a* với một tên khác và thực hiện sửa (theo câu *b*) trên tệp sao này.
- Theo dõi và uốn nắn học sinh hình thức viết chương trình sao cho thể hiện được tính cấu trúc của nó.

***Về bài 2***

- Chương trình ở câu *a* nhằm cung cấp cho học sinh thuật toán tìm phần tử lớn nhất (nhỏ nhất) trong một mảng, đây là thuật toán hay gặp trong nhiều bài toán thực tế. Trọng tâm của bài này là giúp học sinh hiểu được đoạn chương trình thể hiện thuật toán đó, việc tạo mảng dữ liệu học sinh đã được thực hành ở bài 1.
- Nên gợi ý để học sinh có thể thấy được chương trình 2*a* thể hiện thuật toán tìm phần tử lớn nhất (mặc dù khác với chương trình ở §11, mục 1*b*, ví dụ 1):

– Tìm hiểu vai trò của biến *j*: Qua câu lệnh đưa kết quả ra màn hình; qua lệnh gán khởi trị cho biến *j* trước khi duyệt từng phần tử của mảng bằng vòng lặp *for-do*; qua câu lệnh *if-then* dùng để kiểm tra phần tử thứ *i* của mảng và lưu trữ chỉ số *i* của phần tử này trong biến *j*.

– Có thể cho học sinh một ví dụ bộ dữ liệu input nhỏ (mảng *A* có khoảng 4 đến 8 phần tử cụ thể) và yêu cầu học sinh mô phỏng việc chạy đoạn chương trình thể hiện thuật toán trên giấy nháp. Đối với học sinh khá giỏi có thể hướng dẫn các em theo dõi sự thay đổi các biến khi chạy đoạn chương trình này (với các công cụ Debug của Pascal).

– Phải cho học sinh nắm được điểm mấu chốt là tại bước lặp thứ *i* trong câu lệnh *for-do*, phần tử *A*[*i*] được so sánh với *A*[*j*], nghĩa là so sánh với phần tử lớn nhất tìm được trong phạm vi các phần tử đã được duyệt qua ở trước bước này (từ phần tử thứ nhất đến phần tử thứ *i*–1). So sánh đó được thực hiện để tìm phần tử lớn nhất trong phạm vi có thêm *A*[*i*]. Như vậy, ở bước lặp thứ *i*, biến *j* luôn lưu trữ chỉ số của phần tử lớn nhất đã tìm được trong phạm vi *A*[1] đến *A*[*i*]. Bởi vậy khi thực hiện xong câu lệnh *for-do*, giá trị của biến *j* chính là chỉ số của phần tử lớn nhất cần tìm.

- Nên hỏi học sinh, sau đó giáo viên kết luận, cách sửa để có chương trình tìm phần tử nhỏ nhất (chỉ cần sửa dấu lớn hơn (>) thành dấu nhỏ hơn (<) , có thể cho học sinh sửa và chạy thử ngay).
- Nếu không sợ thiếu thời gian, nên yêu cầu học sinh sửa lại một chỗ trong chương trình để kết quả đưa ra phần tử lớn nhất với chỉ số lớn nhất (trong trường hợp có nhiều phần tử có cùng giá trị lớn nhất đó). Chúng ta mong đợi học sinh sẽ sửa câu lệnh

```
if A[i] > A[j] then j:= i;
```

thành câu lệnh

```
if A[i] >= A[j] then j:= i;
```

- Để đáp ứng yêu cầu ở câu *b*, giáo viên có thể dùng một số câu hỏi gợi ý như:

– Chương trình có cần giữ lại đoạn tìm phần tử lớn nhất không?

– Để đưa ra tất cả các chỉ số của các phần tử đạt giá trị lớn nhất đó có cần duyệt lại tất cả các phần tử trong mảng không?

– Những chương trình có được ở bài 1 cũng đã từng duyệt qua tất cả các phần tử của một mảng, mỗi phần tử duyệt đến đã được kiểm tra theo một điều kiện để quyết định một xử lí liên quan đến phần tử này. Điều đó có gợi cho em cách giải quyết câu *b* (bài 2) hay không?

- Lưu ý rằng, việc tạo mảng có phần tử là các số ngẫu nhiên tạo điều kiện cho học sinh chạy được chương trình với kích thước mảng tương đối lớn.

Muốn học sinh không mất thời gian nhập dữ liệu input để quan sát và tìm hiểu chương trình, giáo viên có thể gợi ý học sinh viết câu lệnh:

```
A[i]:= random(300)-random(300);
```

thay cho hai câu lệnh

```
write('Phan tu thu ',i,' = ');
readln(A[i]);
```