BÀI TẬP VÀ THỰC HÀNH 5

A. Mục đích, yêu cầu

- Củng cố cho học sinh những hiểu biết về kiểu xâu.
- Tổ chức những hoạt động trong phòng máy để học sinh có được các kĩ năng cơ bản làm việc với xâu trong lập trình. Cụ thể là:

– Khai báo kiểu xâu.

– Nhập dữ liệu cho xâu, đưa ra màn hình giá trị của xâu.

– Duyệt qua tất cả các kí tự của xâu để thực hiện xử lí tương ứng với từng kí tự đó.

– Sử dụng được các hàm và thủ tục chuẩn đã trình bày ở sách giáo khoa.

- Cung cấp cho học sinh một vài thuật toán cơ bản và đơn giản thường gặp khi xử lí văn bản, rèn luyện một số kĩ năng cài đặt:

– Tạo một xâu mới từ một xâu ban đầu.

– Đếm số lần xuất hiện của một chữ cái trong một văn bản.

– Tìm kiếm và thay thế sự xuất hiện một từ bằng một từ khác trong toàn bộ văn bản.

- 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

Về bài 1

- Nên cho ví dụ hoặc yêu cầu học sinh cho ví dụ về xâu là palindrome và xâu không là palindrome trước khi học sinh tìm hiểu chương trình đã cho.
- Trong tiết học (lí thuyết) về kiểu xâu, đã có hai ví dụ về việc tạo một xâu mới từ một xâu ban đầu. Đây là điều kiện thuận lợi để học sinh hiểu được

chương trình cho sẵn. Chương trình này linh hoạt duyệt xâu ban đầu theo thứ tự ngược lại (từ cuối xâu quay ngược về đầu xâu) nên tạo được xâu đảo ngược của xâu ban đầu và giải quyết được bài toán đặt ra (kiểm tra xâu có là palindrome hay không).

- Với yêu cầu viết lại chương trình không dùng biến xâu *p*, cần gợi ý cho học sinh khai thác khả năng tham chiếu đến từng kí tự trong xâu thông qua vị trí của kí tự này. Như vậy, không cần thiết phải tạo một xâu mới để cuối cùng so sánh hai xâu, mà chỉ cần so sánh các cặp kí tự ở vị trí đối xứng nhau để kết luận xâu có là palindrome hay không. Nên dùng một số câu hỏi hướng dẫn học sinh cài đặt chương trình như:

– Hãy nhận xét về các cặp kí tự ở vị trí đối xứng nhau trong một xâu palindrome.

– Kí tự ở đầu xâu đối xứng với kí tự cuối xâu, kí tự thứ hai trong xâu đối xứng với kí tự đứng ngay trước kí tự cuối xâu, kí tự ở vị trí *i* đối xứng với kí tự ở vị trí nào?

– Phải so sánh bao nhiêu cặp kí tự trong xâu để biết được xâu đó có là palindrome hay không? Hãy dùng vòng lặp để thực hiện các phép so sánh này.

– Chỉ cần phát hiện được một cặp kí tự ở vị trí đối xứng nhau nhưng khác nhau là đủ kết luận xâu đó không là palindrome. Bởi vậy có thể dùng một biến lôgic để ghi nhận sự phát hiện này. Trước vòng lặp thực hiện các so sánh nói trên, cần khởi trị cho biến lôgic đó, ở mỗi bước lặp, nếu hai kí tự được so sánh khác nhau thì biến lôgic đó sẽ phải thay đổi giá trị. Có thể viết đoạn chương trình thể hiện ý tưởng này ra sao?

Sau đây là một chương trình đáp ứng được yêu cầu thêm của bài 1:

```
var i,x: byte;
    a: string;
    palin: boolean;
begin
    write('Nhap vao xau: ');
    readln(a);
    x:=length(a); {xac dinh do dai cua xau }
    palin:=true; {khoi tao palin, tam coi xau a la palindrome}
    for i:=1 to x div 2 do {so sanh cap ki tu doi xung}
        if a[i]<> a[x-i+1] then palin:=false;
    if palin then writeln('Xau la palindrome')
```

```
      else writeln('Xau khong la palindrome');
   readln
end.
```

- Một số học sinh có thể không dùng vòng lặp *for-do* mà dùng *while-do* và có thể không cần dùng biến lôgic. Một chương trình như vậy vẫn có thể đáp ứng được yêu cầu, chẳng hạn như chương trình sau:

```
var i,x:byte;
   a:string;
begin
   write('Nhap vao xau: '); readln(a);
   x:= length(a); {xac dinh do dai cua xau}
   i:= 1;
   while (i<=(x div 2)) and (a[i]=a[x-i+1]) do i:= i+1;
   if i>(x div 2) then writeln('Xau la palindrome')
      else writeln('Xau khong la palindrome');
   readln
end.
```

Về bài 2

- Nếu chỉ sử dụng kiến thức trình bày trong SGK thì lời giải có thể hơi dài. Ta thấy cần ghi nhận số lần xuất hiện của từng chữ cái. Có tất cả 26 chữ cái '*A*' ... '*Z*'. Có thể dùng mảng một chiều với các phần tử được đánh chỉ số từ 1 đến 26 dùng để ghi nhận số lần xuất hiện trong xâu *S* của các kí tự (tương ứng) từ '*A*' đến '*Z*'. Cụ thể, để ghi nhận số lần xuất hiện của kí tự '*A*' (hay kí tự '*a*', vì không phân biệt chữ hoa với chữ thường), ta có thể dùng *dem*[1]. Để ghi nhận số lần xuất hiện của kí tự '*B*' (hay kí tự '*b*'), ta có thể dùng *dem*[2]. Trong TP có thể dùng hàm *ord*(*ch*) cho giá trị là mã ASCII thập phân của kí tự là giá trị của *ch*. Giáo viên có thể gợi ý và dẫn dắt cho học sinh thấy:

  ord('*A*') – *ord*('*A*') + 1 = 1

  ord('*B*') – *ord*('*A*') + 1 = 2

  ...

  ord('*Z*') – *ord*('*A*') + 1 = 26.

- Ngược lại, từ chỉ số *i* của mảng có thể biết phần tử thứ *i* ghi nhận số lần xuất hiện của kí tự nào, bằng cách dùng hàm *chr*(*X*). Hàm *chr*(*X*) cho giá trị là kí tự có mã ASCII thập phân bằng giá trị của biến *X* kiểu byte hoặc số nguyên *X* trong phạm vi 0..255. Như vậy, *chr*(*i* + *ord*('*A*') – 1) là kí tự được đếm bằng phần tử thứ *i* của mảng *dem*.

- Giải quyết vấn đề không phân biệt chữ hoa hay chữ thường bằng cách dùng hàm *upcase*(*c*).
- Do một kí tự xuất hiện trong xâu *S* có thể không phải là một chữ cái nên khi duyệt lần lượt từng kí tự trong xâu *S*, cần kiểm tra xem kí tự đó có phải là chữ cái hay không để ghi nhận số lần xuất hiện của nó. Học sinh đã từng gặp đoạn chương trình kiểm tra một kí tự có là chữ số hay không (trong ví dụ 5 mục 3), từ đó có thể viết được đoạn chương trình thực hiện duyệt từng phần tử của xâu và đếm. Khi duyệt đến kí tự thứ *i* trong xâu *S*, nếu *s*[*i*] là chữ cái thì đếm tăng cho *s*[*i*], nghĩa là tăng 1 cho *dem*[*ord*(*s*[*i*]) − *ord*('*A*') + 1].
- Tuỳ theo trình độ học sinh, giáo viên có thể đưa ra một dàn ý chương trình và yêu cầu học sinh chi tiết hoá bằng các câu lệnh để có một chương trình chạy đúng. Dàn ý chương trình có thể được đưa ra trong giờ thực hành hoặc đưa cho học sinh chuẩn bị ở nhà trước buổi thực hành. Sau đây là ví dụ về một dàn ý chương trình:

```
{phần khai báo}
begin
    {Nhập xâu S}
    N:=length(S);
    {Khởi trị cho mảng dem}
    for i:=1 to N do
        {Nếu s[i] là chữ cái thì đếm tăng cho s[i]}
    for i:=1 to 26 do
        {Thông báo số lần xuất hiện của chr(i+ord('A')-1)}
end.
```

- Một phương án khác, giáo viên có thể hướng dẫn dùng một mảng với chỉ số là kí tự từ '*A*' đến '*Z*' để ghi nhận số lần xuất hiện của các kí tự trong xâu *S*. Khi đó, dàn ý chương trình có thể như sau:

```
{phần khai báo}
begin
    {Nhập xâu S}
    N:=length(S);
    {Khởi trị cho mảng dem}
    for i:=1 to N do
        {Nếu s[i] là chữ cái thì đếm tăng cho s[i]}
    for c:='A' to 'Z' do
        {Thông báo số lần xuất hiện của c}
end.
```

Tuy nhiên, phương án này chỉ dành cho các học sinh khá, giỏi, do sách giáo khoa không trình bày kiểu chỉ số khác với đoạn số nguyên liên tiếp.

Về bài 3

- Để thay thế tất cả cụm kí tự "*anh*" trong xâu *st* thành cụm kí tự "*em*", có thể làm một cách tự nhiên: Tìm vị trí xâu con "*anh*" trong xâu *st* đã cho, xoá xâu con này đi rồi chèn xâu "*em*" vào vị trí đó. Lặp đi lặp lại điều này cho đến khi không tìm thấy xâu "*anh*" cần thay thế trong xâu *st* nữa. Có thể hỏi học sinh rằng với các hàm và thủ tục chuẩn đã biết (đối với kiểu xâu) có thể làm được những việc đó hay không (việc tìm vị trí xuất hiện một xâu con, xoá một xâu con, chèn một xâu con), câu hỏi này làm học sinh phải nhớ lại và vận dụng những hiểu biết về hàm *pos*, các thủ tục chuẩn *delete*, *insert*.
- Giáo viên có thể cùng học sinh thống nhất một dàn ý chương trình và yêu cầu học sinh chi tiết hoá bằng các câu lệnh để có một chương trình chạy đúng. Dàn ý chương trình có thể được đưa ra trong giờ thực hành hoặc cho học sinh chuẩn bị ở nhà trước giờ thực hành. Ví dụ về một dàn ý chương trình:

  {*phần khai báo*}

  begin

  {*nhập xâu S*}

  {*chừng nào còn tìm thấy xâu con "anh" trong xâu S còn làm ba việc sau*:

  – Tìm vị trí bắt đầu của xâu "*anh*";

  – Xoá xâu "*anh*" vừa tìm thấy;

  – Chèn xâu "*em*" vào xâu *S* tại vị trí trước đây xuất hiện xâu "*anh*"}.

  {*in xâu S kết quả*}

  end.

Giáo viên mong đợi học sinh chi tiết hoá dàn ý trên để có được chương trình như sau:

```
var vt: byte;
    st: string;
begin
   write('Nhap vao mot xau: ');
```

```
    readln(st);
    while pos('anh',st)<>0 do
       begin
          vt:= pos('anh',st);
          delete(st, vt, 3);
          insert('em', st, vt);
       end;
    write(st);
    readln
end.
```