VẬN DỤNG MẢNG ĐÁNH DẤU TRONG VIỆC GIẢNG DẠY MÔN KỸ THUẬT LẬP TRÌNH

I. THỰC TRẠNG

Môn Kỹ thuật Lập trình là môn học tích hợp nằm trong khối kiến thức cơ sở của ngành Công nghệ thông tin. Bên cạnh việc cung cấp các kiến thức cơ bản về ngôn ngữ lập trình C/C++ và kỹ năng lập trình cơ bản thì một mục tiêu quan trọng không thể thiếu được là phải rèn luyện cho HS-SV khả năng tư duy thuật toán và chọn lựa giải thuật tối ưu để giải quyết các bài toán lập trình.

Thực tế cho thấy nhiều GV quan niệm là bậc trung cấp và cao đẳng thì chỉ cần dạy SV biết lập trình nên không chú ý rèn luyện cho HS-SV khả năng tư duy thuật toán hay thói quen cải tiến hay tối ưu chương trình. Trong khi các yếu tố này rất cần thiết đối với các lập trình viên chuyên nghiệp. Chính vì thế mà hầu hết các kỳ thi về lập trình hiện nay đều chủ yếu tập trung vào khả năng tư duy thuật toán và tối ưu chương trình. Ngay cả các công ty khi tuyển dụng lập trình viên đặc biệt là lập trình viên backend cũng thường có bài test đầu tiên là để kiểm tra khả năng tư duy thuật toán của ứng viên.

II. NỘI DUNG SÁNG KIẾN

Để rèn luyện khả năng tư duy thuật toán cho HS-SV không nhất thiết phải sử dụng các bài tập phức tập phức tạp mà cần bắt đầu từ những bài tập đơn giản hay các ví dụ trong bài giảng. Quan trọng là GV cần ý thức dạy cho HS-SV cách tư duy để tìm ra thuật toán cũng như ý thức cải tiến và tối ưu thuật toán.

Theo kinh nghiệm của tác giả, với đối tượng HS-SV bậc Trung cấp và Cao đẳng của Trường thì một trong các kỹ thuật tối ưu thuật toán mà GV có thể sử dụng và lồng nghép trong bài giảng môn Kỹ Thuật Lập trình là kỹ thuật dùng **"mảng đánh dấu".**

1. Giới thiệu kỹ thuật dùng mảng đánh dấu

- Kỹ thuật dùng mảng đánh dấu là một trong các kỹ thuật giúp tối ưu chương trình đơn giản và hiệu quả nên được xem là một kỹ thuật cần phải biết đối với các lập trình viên backend chuyên nghiệp và thường được sử dụng trong các cuộc thi lập trình lớn như Kỳ thi Lập trình SV quốc tế ICM/ICPC, Olympic Tin học SV Toàn quốc hay các cuộc thi tuyển chọn lập trình viên của các công ty công nghệ lớn.

- Nội dung của kỹ thuật dùng mảng đánh dấu là dùng một mảng để đánh dấu/ghi nhận lại các số/giá trị cần lưu trữ, tính toán trong lập trình nhằm giảm thời gian xử lý của chương trình hoặc giảm bộ nhớ cần sử dụng.

- Do ý tưởng của kỹ thuật dùng mảng đánh dấu khá đơn giản nên rất phù hợp để GV lồng ghép trong quá trình giảng dạy môn Kỹ thuật Lập trình đặc biệt là đối với bậc cao đẳng và trung cấp.

2. Các trường hợp vận dụng mảng đánh dấu hiệu quả và bài tập minh họa

Trong môn Kỹ thuật Lập trình, có thể vận dụng kỹ thuật dùng mảng đánh dấu trong các bài tập ở các chương sau:

- Chương 5: Mảng và chuỗi

- Chương 7: Kiểu dữ liệu do người dùng định nghĩa

Theo kinh nghiệm của tác giả, trong khuôn khổ môn Kỹ thuật Lập trình thì các trường hợp vận dụng kỹ thuật dùng mảng đánh dấu rất hiệu quả, giúp giảm thao tác xử lý tính toán hay tiết kiệm bộ nhớ trong lập trình là :

TRƯỜNG HỢP 1: Kiểm tra người dùng nhập dãy số nguyên không trùng nhau.

Bài tập 1: Viết chương trình cho phép người dùng nhập n số nguyên không âm nhỏ hơn 100 không trùng nhau (Nếu người dùng nhập trùng thì yêu cầu nhập lại).

<u>Cách thông thường:</u>

- Lưu từng số người dùng nhập vào mảng a gồm n phần tử

- Với mỗi số a[i] được nhập thì duyệt từ đầu mảng để kiểm tra xem số này có trùng với các số từ a[0] đến a[i-1] không ? Nếu trùng thì yêu cầu người dùng nhập lại.

Cách dùng mảng đánh dấu:

- Tạo mảng a có 100 phần tử ban đầu gồm toàn số 0.
- Với mỗi số x được nhập thì kiểm tra nếu a[x]≠0 thì yêu cầu người dùng nhập lại, ngược lại tăng a[x] lên 1.

So sánh 2 giải thuật:

Cách thông thường	**Cách dùng mảng đánh dấu**
Phải sử dụng 2 vòng lặp lồng nhau (lặp n(n+1)/2 lần)	Chỉ sử dụng 1 vòng lặp (lặp n lần)
Độ phức tạp tính toán là $O(n^2)$	Độ phức tạp tính toán là $O(n)$

TRƯỜNG HỢP 2: Sắp xếp dãy số nguyên

Bài tập 2: Sắp xếp dãy số nguyên không âm $a_0, a_1,….,a_{n-1}$ tăng dần với $n \leq 10^5$, a[i] <1000.

Trong chương trình môn Kỹ thuật Lập trình SV chưa được học các giải thuật sắp xếp nên với các bài tập mà khi thực hiện cần phải sắp xếp dãy số thì thường GV sẽ dạy cho SV hàm sắp xếp đơn giản theo phương pháp đổi chỗ trực tiếp (Interchange Sort) có độ phức tạp $O(n^2)$ hoặc sử dụng hàm sort() có sẵn trong C có độ phức tạp O(nlogn).

Cách thông thường:

- Lưu trữ dãy số nguyên vào mảng a tối đa 10^5 phần tử.

Nếu đề mở rộng cho n lớn hơn, ví dụ $n=10^9$ thì cách này sẽ phá sản vì không thể khai báo mảng

- Dùng hàm sắp xếp trên mảng a.

Giải thuật sắp xếp nhanh nhất có độ phức tạp O(nlogn).

Cách dùng mảng đánh dấu:

- Khởi tạo mảng a có 1000 phần tử ban đầu bằng 0.

- Đánh dấu các giá trị trong dãy số nguyên vào a theo chỉ số của mảng bằng cách: với mỗi số x được nhập thì tăng a[x] lên 1.

Ví dụ dãy : 2, 4, 3, 1, 4,… sẽ được lưu trữ như sau:

i	0	1	2	3	4	…	1000
a[i]	0	1	1	1	2	…	

- Khi đó chỉ cần đọc các vị trí có a[i]>0 ta được dãy tăng dần: 1, 2, 3, 4, 4

Nếu a[i]<0 vẫn dùng được cách này nhưng phải tịnh tiến vị trí lưu trữ

So sánh 2 giải thuật:

Cách thông thường	**Cách dùng mảng đánh dấu**
Phải dùng mảng có 100.000 phần tử	Chỉ dùng mảng có 1.000 phần tử
Phải sử dụng hàm sắp xếp	Không cần sử dụng hàm sắp xếp
Độ phức tạp tính toán là O(nlogn)	Độ phức tạp tính toán là O(n)

TRƯỜNG HỢP 3: Tính toán liên quan đến số lần xuất hiện của các số nguyên trong dãy số

Bài tập 3: Cho dãy n số nguyên không âm nhỏ hơn 100 ($n \leq 10^5$). Xuất ra số lần xuất hiện của từng số nguyên trong dãy số.

Cách thông thường:

- Lưu trữ dãy số nguyên vào mảng a tối đa 10.000 phần tử.

- Dùng hàm sắp xếp trên mảng a.

- Duyệt mảng đã sắp xếp và để đếm số lần xuất hiện của từng số nguyên trong mảng và xuất kết quả.

Cách dùng mảng đánh dấu:

- Tạo mảng a có 100 phần tử ban đầu gồm toàn số 0
- Đánh dấu các giá trị trong dãy số nguyên vào a theo chỉ số của mảng bằng cách: với mỗi số x được nhập thì tăng a[x] lên 1.
- Duyệt mảng a để xuất ra a[i] là số lần xuất hiện của số nguyên có giá trị i.

So sánh 2 giải thuật:

Cách thông thường	**Cách dùng mảng đánh dấu**
Phải dùng mảng có 100.000 phần tử	Chỉ dùng mảng có 100 phần tử
Phải sử dụng hàm sắp xếp	Không cần sử dụng hàm sắp xếp
Phải đếm số lần xuất hiện của từng số nguyên	Không cần đếm số lần xuất hiện của từng số nguyên
Độ phức tạp tính toán là O(nlogn)	Độ phức tạp tính toán là O(n)

Một số bài tập khác liên quan đến số lần xuất hiện của các số nguyên có thể áp dụng cách làm tương tự như bài tập 3:

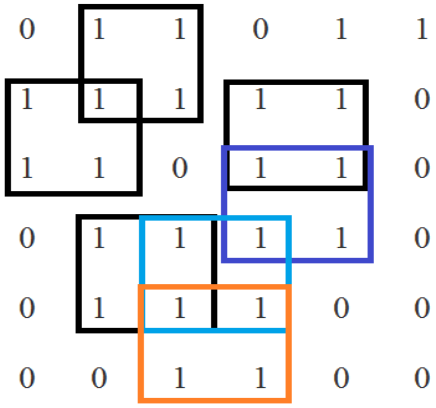
- Tìm số xuất hiện nhiều nhất trong dãy số.
- Tìm số nguyên không xuất hiện trong dãy số.
- Đếm xem có bao nhiêu số nguyên khác nhau.

TRƯỜNG HỢP 4: Vận dụng mảng đánh dấu để tiết kiệm bộ nhớ lưu trữ.

Bài tập 4: (Mô phỏng đề thi khối không chuyên kỳ thi Olympic Tin học SV Toàn quốc năm 2005) Cho ma trận kích thước n x n ($10 \le n \le 700$) với mỗi ô (I,J) ($1 \le I, J \le n$) có giá

trị 0 hoặc 1. Đếm số hình vuông kích thước n x n ($1 \le n \le 40$) mà tất cả các giá trị trong hình vuông đều phải khác 0.

Ví dụ trong ma trận sau với n=6, k=2 ta đếm được 7 hình vuông



Cách thông thường:

- Lưu trữ ma trận vào mảng 2 chiều a.

- Duyệt từng phần tử của ma trận, với mỗi phần tử dương kiểm tra xem có tồn tại hình vuông kxk bắt đầu từ điểm này hay không ?

Cách này sẽ phá sản nếu số n lớn hơn, ví dụ n>=800.

Code chương trình

```
int main()
{
    FILE *f1,*f2;
    f1=fopen("DULIEU.INP","r");
    f2=fopen("DULIEU.OUT","w");
    int a[700][700],n,k,KQ=0,i,j,x,y;
    fscanf(f1,"%d%d",&n,&k);
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
            fscanf(f1,"%d",&a[i][j]);
    }

    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
```

```
        {
            for(x=0;x<k;x++)
            {
                for(y=0;y<k;y++)
                    if(a[i+x][j+y]==0)
                        break;
                if (y<k) break;
            }
            if(x==k) KQ++;
        }
    }
    fprintf(f2,"%d",KQ);
    fclose(f1);
    fclose(f2);
    return 0;
}
```

Cách dùng mảng đánh dấu:

- Đọc từng phần tử của ma trận và xử lý luôn trong quá trình đọc mà không cần lưu trữ ma trận.

- Dùng mảng 1 chiều a để đánh dấu và lưu giá trị tính toán trung gian.

- Với mỗi dòng dữ liệu dùng biến dem để đếm xem có bao nhiêu số 1 liên tục nhau.

Nếu tại vị trí i mà dem ≥ k thì tăng a[i] lên 1 đơn vị và kiểm tra : nếu a[i]>=k thì tăng biến KQ đếm số hình vuông lên 1, ngược lại cho a[i]=0.

Code chương trình:

```
int main()
{
    FILE *f1,*f2;
    f1=fopen("DULIEU.INP","r");
    f2=fopen("KETQUA.OUT","w");
    int a[800]={0},n,k,dem,KQ=0,t,i,j;
    fscanf(f1,"%d%d",&n,&k);
    for(i=0;i<n;i++)
    {
        dem=0;
        for(j=0;j<n;j++)
        {
            fscanf(f1,"%d",&t);
            if(t>0) dem++;
            else dem=0;
```

```
            if(dem>=k)
            {
                a[j]++;
                if(a[j]>=k) KQ++;
            }
            else a[j]=0;
        }
    }
    fprintf(f2,"%d",KQ);
    fclose(f1);
    fclose(f2);
    return 0;
}
```

Ví dụ: Bảng sau mô tả giá trị của mảng a sau khi đọc dữ liệu của từng dòng

Dòng	a[0]	a[1]	a[2]	a[3]	a[4]	a[5]
0	0 (d=0)	0 (d=1)	1 (d=2)	0 (d=0)	0 (d=1)	1 (d=2)
1	0 (d=1)	1 (d=2)	2 (d=3, KQ=1)	1 (d=3)	1 (d=4)	0 (d=0)
2	0 (d=1)	2 (d=2, KQ=2)	0 (d=0)	0 (d=1)	2 (d=2, KQ=3)	0 (d=0)
3	0 (d=0)	0 (d=1)	1 (d=2)	1 (d=3)	3 (d=4, KQ=4)	0 (d=0)
4	0 (d=0)	0 (d=1)	2 (d=2, KQ=5)	2 (d=3, KQ=6)	0 (d=0)	0 (d=0)
5	0 (d=0)	0 (d=0)	0 (d=1)	3 (d=2, KQ=7)	0 (d=0)	0 (d=0)

So sánh 2 giải thuật:

Cách thông thường	**Cách dùng mảng đánh dấu**
Dùng mảng 2 chiều với 700x700 phần tử (Nếu số n lớn thì không khai báo được do đầy bộ nhớ)	Chỉ dùng mảng một chiều với 700 phần tử
Dùng 4 vòng lặp lồng nhau để đếm số hình vuông	Chỉ dùng 2 vòng lặp lồng nhau để đếm số hình vuông
Độ phức tạp tính toán là $O(n^2xk^2) \approx O(n^3)$	Độ phức tạp tính toán là $O(n^2)$

3. Vận dụng mảng đánh dấu trong việc lập trình các chò trơi và bài toán mang tính thực tế

Bài tập 5: Viết chương trình Trò chơi Sổ số Vietlott (bài tập chương 5)

Viết chương trình cho người chơi đoán 6 số trong phạm vi từ 1 đến 45. Sau đó chương trình sẽ sinh 6 số ngẫu nhiên trong phạm vi 1 đến 45.

Nếu người chơi đoán đúng cả 6 số: Xuất thông báo "Bạn đã trúng 12 tỷ đồng"

Nếu người chơi đoán đúng 5 số: Xuất thông báo "Bạn đã trúng 10 triệu đồng"

Nếu người chơi đoán đúng 4 số: Xuất thông báo "Bạn đã trúng 300 ngàn đồng"

Nếu người chơi đoán đúng 3 số: Xuất thông báo "Bạn đã trúng 30 ngàn đồng"

Nếu người chơi đoán đúng ít hơn 3 số: Xuất thông báo số con số đoán đúng và "Chúc bạn may mắn lần sau".

```
          CHUONG TRINH XO SO VIETLOTT

          Moi ban chon 6 so tu 1 den 45
So thu 1: 7

So thu 2: 2

So thu 3: 15

So thu 4: 30

So thu 5: 42

So thu 6: 21

Day so ban chon: 2   7   15   21   30   42

-------------------------------------------

Ket qua xo so:   4   9   11   15   21   29

-------------------------------------------

Ban doan trung 2 so. Chuc ban may man lan sau!
```

Chương trình gồm các phần sau:

- Cho người chơi chọn 6 số, kiểm tra nếu số chọn sau trùng với số đã chọn thì yêu cầu. chọn số khác.

- Cho hệ thống sinh số ngẫu nhiên 6 số, kiểm tra nếu số sinh sau trùng với số đã sinh trước thì sinh số khác.

- Xuất dãy số đã chọn của người dùng và dãy số trúng giải theo thứ tự tăng dần.

- Đếm xem người dùng đoán trúng bao nhiêu số và xuất kết quả.

<u>Cách thông thường:</u>

```
void HoanVi(int &a, int &b)
{
```

```
    int t=a;
    a=b;
    b=t;
}
void SapXep(int a[], int n)
{
    for (int i=0;i<n-1;i++)
        for (int j=i+1;j<n;j++)
            if (a[i]>a[j])
                HoanVi (a[i],a[j]);

}
int main()
{
    int a[6],b[6],i,j,dem=0;
    printf("\t\t\tCHUONG TRINH XO SO VIETLOTT\n");
    printf("\t\t\tMoi ban chon 6 so tu 1 den 45:\n");

    for (i=0;i<6;i++)
    {
        printf("So thu %d: ",i+1);
        scanf("%d",&a[i]);
        for (j=0;j<i;j++)
            if (a[i]==a[j]) break;
        if (j<i||a[i]<1||a[i]>45) i--;
    }

    srand((int)time(0));
    for (i=0;i<6;i++)
    {
        b[i]=rand()%45+1;
        for (j=0;j<i;j++)
            if (b[i]==b[j]) break;
        if (j<i) i--;
    }

    SapXep(a,6);
    SapXep(b,6);

    printf("\n-----------------------------------------------\n");
    printf("Day so ban chon:\t");
    for (i=0;i<6;i++)
        printf("%d\t",a[i]);

    printf("\n-----------------------------------------------\n");
    printf("\nKet qua xo so:");
    for (i=0;i<6;i++)
```

```
        printf("%d\t",b[i]);
    for (i=0;i<6;i++)
        for (j=0;j<6;j++)
            if (a[i]==b[j])
            {
                dem++; break;
            }

    printf("\nBan doan trung %d so. ", dem);
    switch (dem)
    {
        case 6:printf("Chuc mung ban trung 12 ty dong !"); break;
        case 5:printf("Chuc mung ban trung 10 trieu dong !");
             break;
        case 4:printf("Chuc mung ban trung 300 ngan dong !");
             break;
        case 3:printf("Chuc mung ban trung 30 ngan dong !"); break;
        default:printf("Chuc ban may man lan sau !");
    }
    getch();
    return 0;
}
```

Cách dùng mảng đánh dấu:

```
int main()
{
    int a[46]={0},b[46]={0},i,j,dem=0,x;
    printf("\t\t\tCHUONG TRINH XO SO VIETLOTT\n");
    printf("\t\t\tMoi ban chon 6 so tu 1 den 45:\n");
    for (i=0;i<6;i++)
    {
        printf("So thu %d: ",i+1);
        scanf("%d",&x);
        if (a[x]||x<1||x>45)
            i--;
        else
            a[x]=1;
    }
    srand((int)time(0));
    for (i=0;i<6;i++)
    {
        x=rand()%45+1;
        if (a[x]==2)
        {
            i--; break;
        }
        if (a[x]==1)
```

```
            dem++;
        a[x]=2;
    }
    printf("\n-----------------------------------------------\n");
    printf("Day so ban chon:\t");
    for (i=1;i<46;i++)
        if (a[i]==1)
            printf("%d\t",i);
    printf("\n-----------------------------------------------\n");
    printf("\nKet qua xo so:\t\t");
    for (i=1;i<46;i++)
        if (a[i]==2)
            printf("%d\t",i);
    printf("\nBan doan trung %d so. ", dem);
    switch (dem)
    {
        case 6:printf("Chuc mung ban trung 12 ty dong !"); break;
        case 5:printf("Chuc mung ban trung 10 trieu dong !");
             break;
        case 4:printf("Chuc mung ban trung 300 ngan dong !");
             break;
        case 3:printf("Chuc mung ban trung 30 ngan dong !"); break;
        default:printf("Chuc ban may man lan sau !");
    }
    getch();
    return 0;
}
```

So sánh 2 giải thuật:

Cách thông thường	**Cách dùng mảng đánh dấu**
Dùng 2 mảng a,b	Chỉ cần dùng 1 mảng a
Dùng 2 vòng lặp lồng nhau để nhập/sinh số và kiểm tra số trùng.	Chỉ dùng 1 vòng lặp để nhập/sinh số và kiểm tra trùng.
Phải sắp xếp 2 mảng a, b.	Không cần sắp xếp mảng.
Dùng 2 vòng lặp lồng nhau để đếm số trúng.	Không cần sử dụng thêm vòng lặp để đếm số trúng.
Độ phức tạp tính toán là $O(n^2)$.	Độ phức tạp tính toán là O(n).

Bài tập 6: (Bài 2, Đề thi Olympic Tin học Sinh viên Toàn quốc, khối Đại học chuyên tin năm 2010)

Một bãi đỗ xe nhận trông xe trong vòng một tháng. Mỗi xe sẽ được gắn một số hiệu là một số nguyên dương T (10102010 ≤ T ≤ 10109999). Hai xe khác nhau sẽ được gắn hai số hiệu khác nhau. Một xe có thể ra vào bãi đỗ xe nhiều lần, mỗi lần vào bãi đỗ xe, người trông xe sẽ ghi vào sổ sách số hiệu của chiếc xe đó. Cuối tháng dựa vào sổ ghi chép, người trông xe làm thống kê về số lần vào bãi đỗ xe của từng chiếc xe để tiến hành thu phí.

Nếu một chiếc xe vào bãi đỗ xe p lần, cuối tháng chủ xe phải trả một lượng phí được tính như sau:

$$C=\begin{cases}100 & ,nếu\ p \le 5\\ 100+(p-5), & nếu\ p > 5\end{cases}$$

Yêu cầu: Tính tổng số phí người trông xe thu được vào cuối tháng.

Dữ liệu: Vào từ file văn bản PARK.INP có dạng:

- Dòng đầu chứa một số nguyên dương K ($0 < K \le 10^6$)

- K dòng tiếp theo, mỗi dòng chứa số hiệu một chiếc xe .

Kết quả: Đưa ra file văn bản PARK.OUT một số nguyên là tổng số phí thu được

PART.INP
7 10102010 10108888 10102010 10102010 10102010 10102010 10102010

PART.OUT
201

Cách thông thường:

- Lưu số hiệu xe vào mảng a.

- Sắp xếp mảng a tăng dần. (Nếu dùng phương pháp tốt nhất thì độ phức tạp tính toán là O(nlogn)

- Duyệt mảng a để đếm số lần xuất hiện của mỗi số hiệu xe và tính tiền thu được.

Cách dùng mảng đánh dấu:

- Khởi tạo mảng a gồm 7990 phần tử ban đầu =0

- Với mỗi số hiệu xe x thì tăng a[x-10102010] lên 1 đơn vị.

- Duyệt mảng a để tính tổng số tiền thu được.

So sánh 2 giải thuật:

Cách thông thường	**Cách dùng mảng đánh dấu**
Phải dùng mảng a với 10^6 phần tử	Chỉ cần dùng mảng a với 7990 phần tử
Phải sắp xếp mảng a.	Không cần sắp xếp mảng a.
Phải đếm số lần xuất hiện của mỗi số hiệu xe.	Không cần đếm số lần xuất hiện của mỗi số hiệu xe vì mảng a đã lưu trữ số lần xuất hiện của từng số hiệu xe.
Độ phức tạp tính toán là O(nlogn)	Độ phức tạp tính toán là O(n).

III. KẾT LUẬN

Việc vận dụng kỹ thuật dùng mảng đánh dấu không những giúp giúp đơn giản hóa bài toán, tối ưu giải thuật, tiết kiệm thời gian chạy chương trình mà trong nhiều trường hợp còn giúp giảm bộ nhớ. Quan trọng hơn là thông qua việc vận dụng kỹ thuật dùng mảng đánh dấu trong việc giảng dạy môn Kỹ thuật Lập trình sẽ giúp SV có thêm một chìa khóa để mở cánh cửa tư duy sáng tạo và đột phá, kích thích sự đam mê lập trình đồng thời giúp cho SV có được khả năng tư duy thuật toán, là yêu cầu rất quan trọng đối với các lập trình viên.