HƯỚNG DẪN CODE CHU TRÌNH EULER, ĐƯỜNG ĐI EULER

Chú ý: Trong hướng dẫn này, chỗ nào có cụm từ "bạn viết code" hay đại loại thế. Thì bạn phải viết code chỗ đó hén.

Khi bạn làm tới phần này, thì bạn **đã làm được việc đọc thông tin** của đồ thị từ một file nào đó vào chương trình của bạn rồi hén. Nếu bạn vẫn chưa làm được điều này thì đề nghị bạn mở lại file **"HƯỚNG DẪN CODE NHẬP XUẤT MA TRẬN KỀ TỪ FILE"** đọc và làm nhé. Còn nếu bạn đã làm được rồi thì chúng ta tiếp tục hén **J**

Nhắc lại: Thông tin đồ thị của bạn sẽ được lưu trữ trong chương trình thông qua một cấu trúc như sau đúng không?

```
#define MAX 10 // định nghĩa giá trị MAX
#define  inputfile  "C:/test.txt" // định nghĩa đường dẫn tuyệt đối đến file chứa thông tin của đồ thị
typedef struct GRAPH {
	int n; // số đỉnh của đồ thị
	int a[MAX][MAX]; // ma trận kề của đồ thị
}DOTHI;
```

Bước 1: Tạo một cấu trúc Stack như sau để lưu lại các đỉnh trong chu trình euler cũng như đường đi euler:

```
struct STACK
{
	int array[100]; // lưu lại thứ tự các đỉnh trong chu trình euler, đường đi euler, có tối đa 100 đỉnh
	int size; // số lượng các đỉnh trong chu trình euler, đường đi euler.
};
```

Để khởi tạo một stack rỗng, ta tạo một hàm khoitaoStack như sau:

```
void khoitaoStack (STACK &stack)
{
	stack.size = 0; // khởi tạo stack thì kích thước của stack bằng 0
}
```

Để đẩy một giá trị value vào stack, ta gọi hàm DayGiaTriVaoStack như sau:

```
void  DayGiaTriVaoStack (STACK &stack, int value)
{
	if(stack.size + 1 >= 100)  // nếu stack đã đầy thì không đẩy giá trị đó vào được vì kích thước của stack chỉ có chứa tối đa 100 phần tử.
		return;  //thoát không thực hiện đẩy giá trị vào stack nữa
	stack.array[stack.size] = value; // đẩy giá trị value vào stack
	stack.size++; // tăng kích thước stack lên
}
```

Hàm **tìm đường đi** từ một đỉnh i đối với đồ thị g, hàm này viết theo dạng đệ qui.

```
void TimDuongDi (int i, DOTHI &g, STACK & stack)
{
	for (int j = 0; j < g.n; j++)
	{
		if (g.a[i][j] != 0) // vì đồ thị vô hướng nên đối xứng do đó chỉ cần kiểm tra g.a[i][j]!=0 thôi, không cần kiểm tra g.a[j][i] != 0
		{
			g.a[i][j] = g.a[j][i] =0; // loại bỏ cạnh nối đỉnh i tới đỉnh j khỏi đồ thị
			TimDuongDi(j,g,stack); // gọi đệ quy tìm đường đi tại đỉnh j
		}
	}
	DayGiaTriVaoStack(stack,i); // đẩy đỉnh i vào trong stack
}
```

Để kiểm tra đồ thị g, có chu trình euler không thì ta gọi hàm kiểm tra chu trình Euler như sau:

```
int KiemTraChuTrinhEuler (DOTHI g)
{
	int i,j;
	int x = 0; // x là giá trị đỉnh bắt đầu xét chu trình euler,  điều kiện x là đỉnh phải có bậc > 0
	/* bạn phải code chỗ này để tìm 1 đỉnh x bắt đầu tìm chu trình euler, đỉnh x này phải có bậc > 0*/
	DOTHI temp = g; //  tạo ra một bản copy của đồ thị để thực hiện thuật toán, vì trong quá trình thi hành thuật toán ta có xóa cạnh nên ta tạo ra bảng copy này để thực hiện việc xóa đó mà không ảnh hưởng đến đồ thị gốc (ban đầu).
	STACK stack; // tạo một stack như thuật toán đã trình bày
	khoitaoStack (stack); // ban đầu stack chưa có gì nên phải khởi tạo nó về 0.
	TimDuongDi(x,temp, stack); // bắt đầu tìm chu trình euler từ đỉnh x trong đồ thị temp, và thứ tự các đỉnh trong chu trình euler được lưu vào stack này.
	/* bạn cần phải kiểm tra xem hàm TimDuongDi có tìm thấy chu trình euler trong đồ thị temp không? Bạn kiểm tra bằng cách nào? Vì đối với thuật toán của mình thì có xóa cạnh (tức cạnh đã đi qua rồi không đi lại được nữa), do đó để làm điều này, đơn giản bạn kiểm tra xem có tồn tại cung hay đường đi nào trong đồ thị temp không? Nếu tồn tại một cung hay cạnh thì đồ thị temp hay đồ thị ban đầu không có chu trình euler và trả về kết quả 0(sai_ tương ứng là không tìm thấy chu trình euler). Bạn viết code kiểm tra điều này hén, đơn giản mà*/
	/* Nếu có chu trình euler thì bắt buộc đỉnh đầu và đỉnh cuối trong stack phải bằng nhau. Điều nay tương đương với việc đỉnh đầu và đỉnh cuối trùng nhau trong chu trình Euler. Nếu đỉnh đầu và đỉnh cuối không trùng nhau thì bạn trả về kết quả 0 (sai_đồ thị temp hay đồ thị ban đầu không có chu trình euler). Bạn viết code kiểm tra điều này hén, quá đơn giản mà*/
		printf("\n Chu Trinh Euler : ");
```

```
        /* Tới bây giờ thì bạn đã có chu trình euler  rồi đó, hãy code mà xuất ra nhé. Dựa vào stack đó đó */
        return 1; // trả về kết quả 1 tức có chu trình euler
        // xong chu trình euler rồi đó. Code hén J
}
```

Bước 2: Trường hợp nếu như đồ thị không tồn tại chu trình euler thì đồ thị đó có tồn tại đường đi euler không???. Để làm điều này tiến hành kiểm tra điều này bằng cách gọi hàm KiemTraDuongDiEuler như sau:

```
int KiemTraDuongDiEuler (DOTHI g)
{
        int i,j;
        int x = 0; // x là giá trị đỉnh bắt đầu xét đường đi euler,  điều kiện x là đỉnh phải có bậc lẽ.
        /* bạn phải code chỗ này để tìm 1 đỉnh x bắt đầu tìm chu trình euler, đỉnh x này phải có bậc lẽ*/
        DOTHI temp = g; //  tạo ra một bản copy của đồ thị để thực hiện thuật toán, vì trong quá trình thi hành thuật toán ta có xóa cạnh nên ta tạo ra bảng copy này để thực hiện việc xóa đó mà không ảnh hưởng đến đồ thị gốc (ban đầu).
        STACK stack; // tạo một stack như thuật toán đã trình bày
        khoitaoStack (stack); // ban đầu stack chưa có gì nên phải khởi tạo nó về 0.
        TimDuongDi(x,temp, stack); // bắt đầu tìm đường đi euler từ đỉnh x trong đồ thị temp, và thứ tự các đỉnh trong đương đi euler được lưu vào stack này.
        /* bạn cần phải kiểm tra xem hàm TimDuongDi có tìm thấy đường đi euler trong đồ thị temp không? Bạn kiểm tra bằng cách nào? Vì đối với thuật toán của mình thì có xóa cạnh (tức cạnh đã đi qua rồi không đi lại được nữa), do đó để làm điều này, đơn giản bạn kiểm tra xem có tồn tại cung hay đường đi nào trong đồ thị temp không? Nếu tồn tại một cung hay cạnh thì đồ thị temp hay đồ thị ban đầu không có đường đi euler và trả về
```

```
kết quả 0(sai_ tương ứng là không tìm thấy đường đi euler). Bạn viết code kiểm tra điều này hén, đơn giản mà*/
        /* Nếu có đường đi euler thì bắt buộc đỉnh đầu và đỉnh cuối trong stack không giống nhau. Điều nay tương đương với việc đỉnh đầu và đỉnh cuối không trùng nhau trong đường đi Euler. Nếu đỉnh đầu và đỉnh cuối trùng nhau thì bạn trả về kết quả 0 (sai_đồ thị temp hay đồ thị ban đầu không có đường đi euler). Bạn viết code kiểm tra điều này hén, quá đơn giản mà*/
        printf("\nĐường đi Euler : ");
        /* Tới bây giờ thì bạn đã có đường đi euler  rồi đó, hãy code mà xuất ra nhé. Dựa vào stack đó đó */
        return 1;
        // xong đường đi euler rồi đó. Code hén J
}
```

Bước 3: Code trong hàm main để gọi hàm các hàm tương ứng và chạy. Có thể làm như sau:

```
void main()
{
        DOTHI g;
        clrscr();
        if (DocMaTranKe(inputfile, g) == 1)
        {
                printf("Da lay thong tin do thi tu file thanh cong.\n\n");
                XuatMaTranKe(g);
                printf("Bam 1 phim bat ki de bat dau xet tim chu trinh euler ...\n\n");
                getch();
                if (!KiemTraChuTrinhEuler(g))
                {
```

```
                printf("Khong co chu trinh Euler trong do thi cua ban\n");
                printf("Bam 1 phim bat ki de bat dau xet tim duong di euler ...\n\n");
                getch();
                if (!KiemTraDuongDiEuler(g))
                {
                        printf("Khong co duong di Euler trong do thi cua ban \n");
                }
        }
    }
    getch();
}
```

HƯỚNG DẪN CHI TIẾT TỚI CỠ NÀY RỒI MÀ BẠN VẪN KHÔNG LÀM ĐƯỢC NỮA THÌ BẠN CHUẨN BỊ TINH THẦN ĐI HÉN J

Chúc các bạn may mắn và học tốt môn này

GOOD LUCK TO U