# NỘI DUNG ÔN TẬP THI KẾT THÚC MÔN

**Phần 1:** Những thao tác liên quan đến DSLK đơn: Xây dựng CTDL cho một bài toán được yêu cầu (số nguyên, phân số, một điểm tròn mặt phẳng OXY, một đa thức, sinh viên, môn học,… ), Thêm (đầu, cuối, trước, sau, bất kỳ, sắp xếp tăng/giảm cho 1 field của dữ liệu có cấu trúc), Xóa (đầu, cuối, trước, sau, bất kỳ), xuất thông tin theo yêu cầu, xử lý tính toán theo yêu cầu.

**Phần 2:** Những thao tác liên quan đến Stack và Queue: Trình bày giải thuật và Ứng dụng Stack để giải quyết một số bài toán thông dụng (đổi cơ số, đảo chuỗi, kiểm tra chuỗi đối xứng, kiểm tra chuỗi palindrome, tính giai thừa, kiểm tra dấu ngoặc hợp lệ, …). Trình bày giải thuật chuyển đổi một biểu thức toán học từ tiền tố (infix) sang hậu tố (postfix), Trình bày giải thuật tính toán giá trị của một biểu thức khi biết dạng hậu tố. Áp dụng cho một biểu thức theo yêu cầu.

**Phần 3:** Những thao tác liên quan đến Cây nhị phân: Xây dựng CTDL cho một bài toán được yêu cầu (số nguyên, phân số, sinh viên,… ), Thêm node, Xóa node, xuất thông tin theo yêu cầu, xử lý tính toán theo yêu cầu.

**Phần 4:** Những thao tác liên quan đến Cây AVL: Xây dựng cây (vẽ cây khi thêm node, xóa node).

# CẤU TRÚC 1 ĐỀ THI

**Câu 1.**

Thông tin của sinh viên (SV) học môn CTDL&GT gồm các thông tin sau:

- Mã số SV là một chuỗi 10 ký tự
- Họ tên sinh viên là một chuỗi 30 ký tự
- Điểm giữa kỳ (GK) là một số thực
- Điểm tiểu luận (TL) là một số thực
- Điểm cuối kỳ (CK) là một số thực

Cho các khai báo cấu trúc dữ liệu của danh sách liên kết đơn lưu trữ thông tin sinh viên như sau:

```
struct SV
{
   char MaSV[11];
   char TenSV[31];
   float GK,TL,CK;
};
struct SNode
{
   SV Info;
   SNode* Next;
};
struct SList
{
   SNode* Head;
   SNode* Tail;
};
```

Giả sử danh sách liên kết lưu trữ một lớp học có n sinh viên (SV). Hãy viết các hàm cho phép xuất ra màn hình thông tin của 3 sinh viên có điểm trung bình môn học cao nhất. Biết rằng điểm trung bình môn học được tính theo công thức: TBM = 20%*TL + 30%*GK +

50%*CK.

Cho trước các khai báo nguyên mẫu hàm như sau:

```
float tinhDTB(SV sv); //trả về điểm TB
void sapXepGiam_DTB(SList &sl); //Sắp xếp danh sách giảm dần theo điểm TB
void inThongTinSV(SV sv); //Xuất thông tin của sinh viên sv ra màn hình
void inThongTin3SV(SList sl); //Xuất thông tin của 3 sinh viên ra màn hình
```

## PHẦN 2: STACK & QUEUE

**Câu 2a:** Trình bày giải thuật chuyển đổi biểu thức dạng trung tố (Infix) có các phép toán +, -, *, /, các toán hạng, dấu ngoặc sang hậu tố (Postfix) sử dụng stack. Hãy mô tả từng bước quá trình chuyển biểu thức trung tố M=(2*3-4+2)/(4-2) sang dạng hậu tố sử dụng Stack. Yêu cầu mô tả từng bước theo dạng bảng sau:

| STT | Ký tự đang đọc | Stack | Biểu thức Hậu tố |
|---|---|---|---|
| | | | |

**Câu 2b:**

a) Trình bày giải thuật tính giá trị của một biểu thức dạng hậu tố có các phép toán +, -, *, /, các số từ 0 tới 9 sử dụng stack.

b) Hãy mô tả từng bước quá trình tính giá trị của biểu thức hậu tố **M = 1 5 * 3 - 2 3 + * 5 /** . Yêu cầu mô tả từng bước theo dạng bảng sau:

| STT | Ký tự đang đọc | Stack | Công việc thực hiện |
|---|---|---|---|
| | | | |

**Câu 2c:**

a) Trình bày giải thuật đổi một số nguyên dương n từ hệ thập phân sang hệ a phân bất kỳ ($2 \le a \le 9$) bằng cách sử dụng stack.

b) Viết hàm đổi một số thập phân n sang hệ a phân bằng cách sử dụng stack. Cho trước khai báo nguyên mẫu hàm như sau:

```
void convertDecimalToA(int n, int a);
```

Hàm này sẽ in ra màn hình biểu diễn hệ a phân tương ứng của số nguyên dương n.

Giả sử đã có cấu trúc Stack lưu các số nguyên và cho trước các khai báo nguyên mẫu hàm sau (*Sinh viên được phép sử dụng mà không cần viết lại code cho các hàm này*):

int **push(Stack s, int x)**: Thêm số nguyên x vào Stack s, hàm trả về 1 nếu thêm thành công, ngược lại trả về 0.

int **pop(Stack s, int &x):** Lấy phần tử ra khỏi Stack s, giá trị lấy được sẽ lưu vào x, hàm trả về 1 nếu lấy ra thành công, ngược lại trả về 0.

int **isEmpty(Stack s)**: kiểm tra Stack s nếu rỗng thì trả về 1 ngược lại trả về 0.

void **initStack(Stack &s):** Khởi tạo Stack s thành stack rỗng.

## PHẦN 3: CÂY NHỊ PHÂN

**Câu 3.** Cho khai báo cấu trúc dữ liệu của cây nhị phân chứa các số nguyên như sau:

```
struct TNode
{
	int Info;
	TNode *Left;
	TNode *Right;
};
```

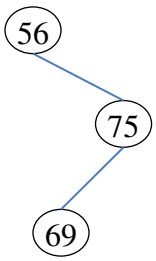
Viết 2 trong số các hàm như: tính chiều cao, đếm số nút, tính tổng, và xuất nội dụng các nút theo yêu cầu (*như ở mức $k \ge 0$*).

## PHẦN 4: CÂY AVL

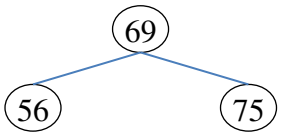
**Câu 4.** Cho một dãy các khóa như sau: 56, 75, 69, 83, 90, 80, 40, 60, 65, 63. Hãy vẽ cây AVL khi thêm lần lượt các giá trị của dãy đã cho vào cây (*vẽ cây minh họa từng bước*).

**4** (2.5)

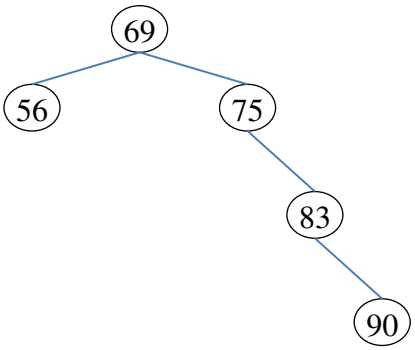
❖ Kết quả khi thêm lần lượt các khóa 56, 75, 69 vào cây như sau:



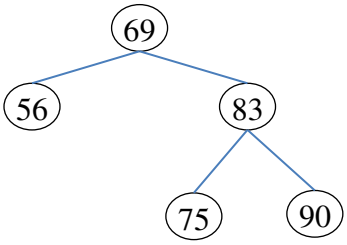

⇨ Cây bị mất cân bằng RL tại 56, vẽ lại cây như sau:



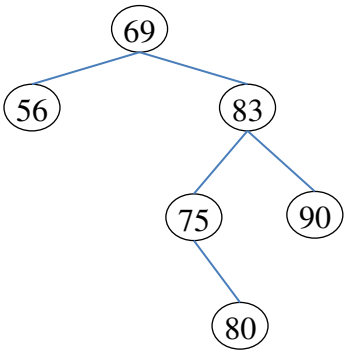

❖ Kết quả khi thêm lần lượt các khóa 83, 90 vào cây như sau:



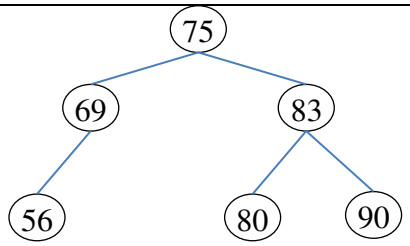

⇨ Cây bị mất cân bằng RR tại 75, vẽ lại cây như sau:



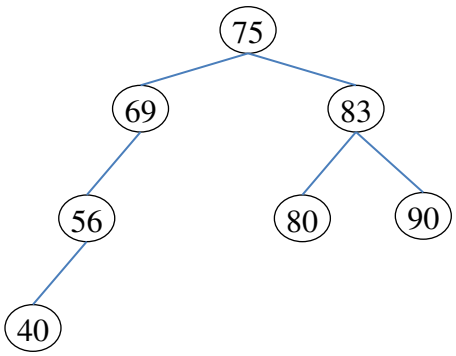

❖ Kết quả khi thêm khóa 80 vào cây như sau:



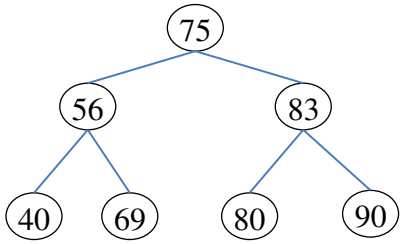

⇨ Cây bị mất cân bằng RL tại 69, vẽ lại cây như sau:



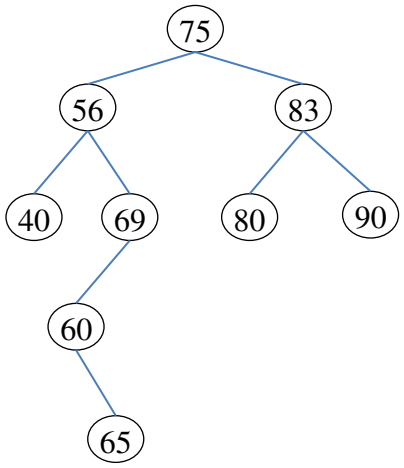

❖ Kết quả khi thêm khóa 40 vào cây như sau:



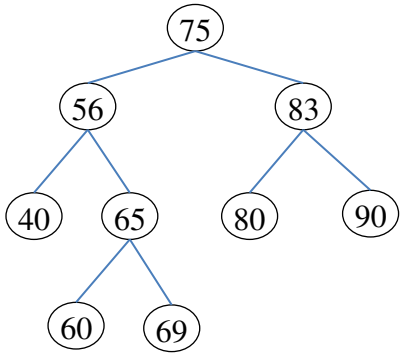

⇨ Cây bị mất cân bằng LL tại 69, vẽ lại cây như sau:



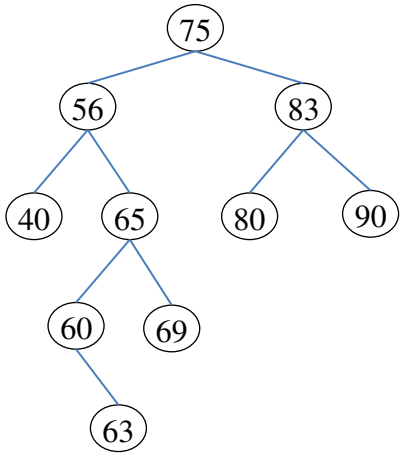

❖ Kết quả khi thêm lần lượt các khóa 60, 65 vào cây như sau:




⇨ Cây bị mất cân bằng LR tại 69, vẽ lại cây như sau:



❖ Kết quả khi thêm khóa 63 vào cây như sau:




⇨ Cây bị mất cân bằng RL tại 56, vẽ lại cây như sau: