

Final - Ca 1

Thời gian làm bài: 70 phút

1 Quy định nộp bài

- Sinh viên phải thực hiện bài kiểm tra tại phòng máy.
- Sinh viên thực hiện các hàm với Prototype có sẵn, trong một file `MSSV.cpp` duy nhất. (Chỉ cần viết hàm, không cần viết hàm `main()`). Nộp `MSSV.cpp`

Lưu ý: Thay cụm MSSV thành mã số sinh viên của bạn.

2 Quy định chấm bài

- Chấm bài bằng trình biên dịch `g++` (thư viện chuẩn)
- Đối với những hàm sinh viên không thực hiện được \rightarrow giữ nguyên trạng thái ban đầu.
- KHÔNG chấm ý tưởng, chỉ có đúng hoặc sai.
- Bài làm ở mỗi phần sẽ chấm theo thứ tự, câu trên sai sẽ không chấm câu dưới liên quan.
- Những trường hợp sau đây sẽ bị 0 điểm bài thi:
 - Nộp sai quy định.
 - BÀI LÀM GIỐNG NHAU.
 - KHÔNG BIÊN DỊCH ĐƯỢC.
 - LẬP VÔ TẬN.

3 Yêu cầu

- **Câu 1** (10 điểm): Cho cây AVL được định nghĩa như sau:

```
struct Node{
    int* a;    //Dynamic array a
    int n;    //size of a
    Node* pLeft;
    Node* pRight;
};
```

– Xét 2 Node **a** và **b**. Node **a** được gọi là lớn hơn Node **b** nếu:

- * Ưu tiên 1: Tổng các phần tử trong **a** lớn hơn tổng các phần tử trong **b**
- * Ưu tiên 2: Nếu 2 Node có tổng các phần tử bằng nhau thì Node **a** lớn hơn Node **b** khi số lượng phần tử của Node **a** nhiều hơn số lượng phần tử của Node **b**

và ngược lại. Cho rằng các input được giới hạn trong 2 ưu tiên nêu trên.

Thực hiện các yêu cầu sau:

- a. Viết hàm đọc thông tin từ file *"inputAVL.txt"* và thêm lần lượt các node vào cây rỗng ban đầu. (2 điểm)

prototype: Node* createTree(string filename)

– Ví dụ: File *"inputAVL.txt"*: 3, {[1, 2, 3]}, {2, [3, 2]}, {1, [5]},
{2, [7, 3]}, {3, [1, 2, 5]}, {4, [1, 2, 3, 4]}

- b. Viết hàm duyệt cây theo level (từ trái sang phải). (2 điểm)

prototype: void levelOrder(Node* pRoot)

– Level 1: 1, 2, 5
– Level 2: 3, 2
– Level 2: 7, 3
– Level 3: 5
– Level 3: 1, 2, 3
– Level 3: 1, 2, 3, 4

- c. Viết hàm xóa 1 node cho sẵn. (2 điểm)

prototype: void remove(Node*& pRoot, int *a, int n)

- **Câu 2:** Xem xét file "*Players.csv*" chứa danh sách các cầu thủ, có cấu trúc mỗi dòng:

Họ và tên, Ngày sinh - Tháng sinh - Năm sinh, Chiều cao, Cân nặng, Quốc tịch, Câu lạc bộ.

Ta cần thực hiện việc lưu trữ các cầu thủ vào bảng băm có kích thước 2500, với hàm băm H được định nghĩa như sau

$$H(d, m, y, T) = d^{\text{sumCha}(T)} \times m^d \times \text{sumDigit}(y)^m \times \text{sumCha}(T)^{\text{sumDigit}(y)} \% M$$

, biết rằng:

- d : ngày sinh của cầu thủ
- m : tháng sinh của cầu thủ
- y : năm sinh của cầu thủ
- T : tên của cầu thủ, chỉ xét các kí tự nằm trong bảng chữ cái ('A' - 'Z' và 'a' - 'z')
- $\text{sumCha}(T) = \sum_{i=0}^{\text{len}(T)-1} (T[i] - 65)$ với $T[i]$ là phần tử thứ i trong chuỗi tên cầu thủ, $T[i]$ mang giá trị của kí tự tương ứng trong bảng mã ASCII.
- $\text{sumDigit}(n)$ là tổng các chữ số của n
- $M = 10^9 + 7$

- a. Viết hàm đọc file "*Players.csv*" và xây dựng bảng băm: (2 điểm)

prototype: `Player* createHash(string filename, int &size)`

- b. Viết hàm in ra thông tin của bảng băm (mỗi dòng cần chứa chỉ số dòng và các thông tin của cầu thủ liên quan): (2 điểm)

prototype: `void printHashTable(Player* &hashTable, int size)`

- c. Viết hàm tìm 1 phần tử trong bảng băm: (1 điểm)

prototype: `Player search(Player* hashTable, Player P)`

- d. Viết hàm xóa 1 phần tử khỏi bảng băm: (1 điểm)

prototype: `void Remove(Player* hashTable, Player P)`