**CPP Coding Problem**

|  |
| --- |
| **Subject: Debugging Code** |
| **Contributor: Yen-Chen Chiu** |
| **Main testing concept: Code Reviewing, working with others.**   |  |  | | --- | --- | | **Basics** | **Functions** | | ■ C++ BASICS  ■ FLOW OF CONTROL  ■ FUNCTION BASICS  ■ PARAMETERS AND OVERLOADING  ■ ARRAYS  ■ STRUCTURES AND CLASSES  □ CONSTRUCTORS AND OTHER TOOLS  □ OPERATOR OVERLOADING, FRIENDS, AND REFERENCES  ■ STRINGS  ■ POINTERS AND DYNAMIC ARRAYS | ■ SEPARATE COMPILATION AND NAMESPACES  □ STREAMS AND FILE I/O  ■ RECURSION  □ INHERITANCE  □ POLYMORPHISM AND VIRTUAL FUNCTIONS  □ TEMPLATES  ■ LINKED DATA STRUCTURES  ■ EXCEPTION HANDLING  □ STANDARD TEMPLATE LIBRARY  ■ PATTERNS AND UML | |
| **Description:**   * **Background Story** (*\*Skippable*):   As a programmer, you always need to debug. Today, your job is to review and debug codes of programs in a project due to your PM (Project Manager) had told you there are some bugs in this program.   * **Brief Introduction**:   In this coding problem, you’d have a program project that can be built and run. But you have to understand how it works, and fix bugs to meet the requirements.   * **Program Guideline**:   This program is a sorting system, which can sort a list of names of guests who are invited by a host or other invited guests. A host can only invite guests, while a guest not only can be invited but can invite more guests as well.  There are 3 types of people: host, VIP guest, and non-VIP guest.  A host or a VIP guest can invite unlimited number of any guests. But a non-VIP guest can only invite up to 3 non-VIP guests, and unlimited number of VIP guests.  Every guest has a priority number (). The priority number determines if the guest is allowed to invite other guests. For a non-VIP guest, its is equal to its inviter’s . (For details, please see the rules below.)   * + **Priority Number /** **Invitations Rules:**  |  |  |  | | --- | --- | --- | | Person type | Priority Number () | How many guests can the person invite? | | Host | -1 | all unlimited. | | VIP Guest | 0 | all unlimited. | | Non-VIP Guest | Inviter’s + 1 |  |  * **Bugs Reported (The Bug List):**   A list of bugs has reported below, you need to fix them all:   1. If I enter the same names again, this program should output “ERROR: This is a duplicated name, cannot be imported.” and do nothing for that duplicated input name. 2. When I tried to add a VIP guest invited by a non-VIP guest who has invited 3 non-VIPs. The system failed to add the VIP guest. But a VIP guest should ALWAYS can be invited. 3. Output list of names is not sorted properly! Eventually, the non-VIP guests should be sorted by their priority numbers first, and then names in alphabetical order.   **Input:**  In this coding program, users should input lines of hosts’ names first, and then the lines of guests’ names. Both ended with a line of “END”.   * + The input format for a host is: “host’s name”   + The input format for a non-VIP guest is: “guest’s name;inviter’s name”.   + The input format for a VIP guest is: “#guest’s name;inviter’s name”.   Note that if a guest is entered with an inviter’s name that has never imported before, the system should not import the guest.  **Output:**   * Before inputting hosts’ names, output these 2 lines:  “Please enter the names of the hosts.” “(Enter "END" when finished):” * After inputting hosts’ names, output these 2 lines: “Please enter the names of the guests and their inviters.” “(Add # in front of the line if it's VIP. Separated with ";". Enter "END" when finished):” * For each name inputted, outputs the import result by these rules:  1. When a person with name *ABC* imported successfully: For host, output: “Host: *ABC* imported.”. For guest invited by *DEF*, output: “Guest: *ABC* (invited by *DEF*) imported.”. For VIP guest invited by *DEF*, output: “Guest(VIP): *ABC* (invited by *DEF*) imported.”. 2. When the name of an inviter doesn’t exist: “ERROR: The inviter doesn't exist.” 3. When the name of a new guest is duplicated: “ERROR: This is a duplicated name, cannot be imported.” 4. When the priority number , cannot invite other non-VIP: “ERROR: The inviter with priority >= 2, can't invite the guest.” 5. When the inviter cannot invite more guests: “ERROR: The inviter can't invite more guests.”   \*If there are multiple errors, output only one message that found first in the order above.   * After all the names are inserted, this system output a list of names sorted by these rules:  1. The hosts’ names would not be on the list. 2. The VIP guests are always on most top of other non-VIP guests. 3. The names of VIP guests should be in alphabetical order. 4. The names of non-VIP guests are sorted by their priority numbers. 5. If some non-VIP guests have same priority number, sorted in alphabetical order.   **Sample Input / Output：**   |  |  | | --- | --- | | Sample Input | Sample Output | | Host A  Host A  Host B  END  ABC;Host A  ABC;Host A  DEF;Host X  DEF;ABC  GHI;DEF  JKL;GHI  #JKL;GHI  PQR;MNO  MNO;Host B  PQR;MNO  STU;MNO  VWX;MNO  #YZA;MNO  BCD;MNO  END | Please enter the names of the hosts.  (Enter "END" when finished):  Host: Host A imported.  ERROR: This is a duplicated name, cannot be imported.  Host: Host B imported.  Please enter the names of the guests and their inviters.  (Add # in front of the line if it's VIP. Separated with ";". Enter "END" when finished):  Guest: ABC (invited by Host A) imported.  ERROR: This is a duplicated name, cannot be imported.  ERROR: The inviter doesn't exist.  Guest: DEF (invited by ABC) imported.  Guest: GHI (invited by DEF) imported.  ERROR: The inviter with priority >= 2, can't invite the guest.  Guest(VIP): JKL (invited by GHI) imported.  ERROR: The inviter doesn't exist.  Guest: MNO (invited by Host B) imported.  Guest: PQR (invited by MNO) imported.  Guest: STU (invited by MNO) imported.  Guest: VWX (invited by MNO) imported.  Guest(VIP): YZA (invited by MNO) imported.  ERROR: The inviter can't invite more guests.  ==============================  The Guest List:  VIP: JKL  VIP: YZA  ABC  MNO  DEF  PQR  STU  VWX  GHI | |
| **□ Easy. Only basic programming syntax and structure are required.**  **□ Medium. Multiple programming grammars and structures are required.**  ■ **Hard. Need to use multiple program structures or complex data types.** |
| **Expected solving time:**  40 minutes |
| **Other notes:**   * It’s OK if you want to change any code. (OJ won’t replace any file.) * Every input/output format remains the same as the program provided.   + Means you only need to fix the bugs reported.   + The input test data won’t contain any bad format.   + The mentioned *alphabetical order* sorting is as same as ASCII order. |