A. 遊戲：外傳

Description

相信有認真寫「計算幾何」加分題作業的你，已經知道空、白兩位天才遊戲玩家，已經多次擊敗了伊綱，並且獲得柴柴、以及伊綱的一個禮拜的僕役。

故事原本到那時就結束了，但是伊綱在當空、白的僕人時，偷偷跟「史蒂芬妮・多拉」學習撲克牌洗牌、記牌的技巧，並且在當僕人的過程中，花了很多時間練習，希望可以藉此打敗空、白。

接下來的故事，會怎麼發展呢？讓我們繼續，看～下～去～

「では、ゲームを始めましょう」。

一天，伊綱跟史蒂芬妮・多拉走在路上，剛好遇到正要前往藥局的空、白。

「讓我們來玩遊戲吧」，伊綱說。「如果我贏了這個遊戲，你們就當我一個月的僕人，です」。

「但是如果我們贏了，妳就要當我們一天的僕人」空、白說。

「盟約に誓って、です」（向盟約發誓，です）。

「盟約に誓って」（向盟約發誓）。

出乎預料的是，伊綱竟然選擇「21 點」當作比賽的遊戲，並且伊綱指定自己當**莊家**，空、白必須當**閒家**。

遊戲規則如下：

- 伊綱會準備 5000 副不含鬼牌的撲克牌（總共有 $52 \times 5000 = 260000$ 張），並且她會均勻的洗牌。
- 空、白一開始會有 10^6 的資金，並且他們每次可以下注任意「正整數數量」的賭注。
- 只要空、白在牌還沒被使用完之前，在某個時刻的資金量高於 1.3×10^6 ，空、白就贏了。
- 否則，只要空、白花超過 5000 副撲克牌，或者是輸光了資金，伊綱就獲勝了。

由於要記憶 52×5000 張牌的資訊有點大，於是 **空、白找上了你**，希望可以藉由你寫的程式，來贏得這場遊戲。

對了，**伊綱為了避免空、白靠賽，她會舉行這個遊戲恰好五次（意謂著測試資料只有五筆），空、白要連續贏五次，才算贏了這個挑戰**。

他們玩的「21 點」規則如下（如果與真實賭場、網路所看到的規則不同，請用以下的規則為主）：

- 牌的點數：
  - $2, 3, 4, \ldots, 10$ 的點數，就是上面數字。(例如，5 就是 5 點。)
  - $J(11), Q(12), K(13)$ 是 10 點
  - A 可以是 1 點，也可以是 11 點。
  - 一副手牌的點數，就是每張牌的點數總和。
  - 如果一副手牌的點數超過 21 點，就是爆牌（busted）。
  - 例如：「K、3、6」是 19 點，「A、5」可以是 16 點，也可以是 6 點、「J、K、2」是爆牌，「A、10、8」是 19 點。
- 牌的大小比較：
  - 如果一副牌是 blackjack ，代表這傅副牌是由「A」跟任意一張 10 點的牌（10、J、Q、K）構成。例如：「A、J」、「Q、A」都是 blackjack。
  - 假設閒家的牌為 X ，莊家的牌為 Y 。
  - 如果 X, Y 都是 blackjack ，他們兩個就平手（push）。
  - 如果 X 是 blackjack 而 Y 不是 blackjack ，那麼閒家就贏了（就算莊家的點數總和是 21 ，也是閒家贏）。
  - 如果 X 不是 blackjack 而 Y 是 blackjack ，那麼莊家就贏了（就算閒家的點數總和是 21 ，也是莊家贏）。
  - 如果 X, Y 都不是 blackjack：
    - 如果 X 爆牌了，那麼莊家就贏了（不管莊家有沒有爆牌，只要閒家爆牌了，就是莊家贏）。
    - 如果 X 沒爆牌，而 Y 爆牌了，那麼閒家就贏了。
    - 否則，就算比較 X, Y 最大的可能點數總和（「A、5」就用 16 點來比較），如果 X 比較大，就是閒家贏，如果 Y 比較大，就是莊家贏。否則，就是平手（push）。
- 在牌局的一開始，空、白（閒家）必須決定下注的金額，假設這個數字是 B。
- 接著，伊綱（莊家）會發兩張牌給閒家，以及兩張牌給莊家。莊家有一張牌是明的（閒家會知道），有一張牌是暗的（只有莊家知道，閒家不知道）。
- 如果莊家的明牌是「A」，這時閒家**必須決定**要不要買保險。買保險的規則如下：
  - 閒家可以決定花 C 塊前買保險。買保險的金額不能高於下注金額的一半（$0 \le C \le \frac{B}{2}$，$C = 0$ 意謂著不買保險）。就算你的牌是 blackjack ，你仍然可以買保險！
  - 如果莊家是 blackjack ，閒家可以拿到 $2C$ 元。下注的 B 塊則拿不回來。這個牌局就到這邊結束了。

  - 如果莊家不是 blackjack ，閒家會喪失買保險的 C 元，並且牌局繼續。

- 接著，閒家可以做的基本操作是加牌（hit）、停止加牌（stand）：
  - 加牌時，莊家會發一張牌給閒家。
  - 如果閒家不想加牌，或者是 **已經爆牌**，就必須停止加牌。
- 當閒家停止加牌時，莊家會把暗牌打開。
- **莊家會一直加牌，直到最大點數不低於 17 點，或是爆牌為止。莊家只能執行前述的策略。**
- 當莊家結束加牌後，會比較莊家、閒家牌的大小，並且根據這個大小關係來決定金錢交易：
  - 如果是「平手」，則閒家拿回當初下注的 B 元。
  - 如果閒家是「blackjack」獲勝，可以拿回 $\frac{5B}{2}$ 元。
  - 如果閒家不是以 blackjack 獲勝，可以拿回 $2B$ 元。
  - 否則，就是莊家贏，閒家拿不到任何的下注。
- 除此之外，還有兩種特殊的規則：加倍（double down）、分牌（split），介紹分別如下：
  - 加倍（double down）：拿到手牌時，**不管點數是多少**，可以決定再付 B 元的賭注，並且從莊家那裡再拿一張牌。之後就是用這三張牌跟莊家比大小。如果你贏了，就拿回總共 $4B$ 的金額，如果平手的話，就拿回 $2B$ ，如果輸的話，什麼都拿不回來。
  - 分牌（split）：**如果一開始的兩張牌點數一樣**，就可以分牌。如果想要執行分牌，必須多付 B 元的賭注。執行分牌時，莊家會把你現有的兩張牌分成兩堆，並且個別補上一張牌。之後這兩堆牌就是獨立的牌堆。閒家必須依序決定這兩堆牌要不要加牌（hit）、停止加牌（stand）。最後跟莊家比較牌的大小時，會把那兩堆當成**獨立**的堆比較。

相信身為資訊之芽算法班學員的你，看過規則之後，就知道要怎麼寫程式了。這題很有趣吧，趕快動手開始寫吧～～～

Instructions

為了模擬真實的遊戲進行，**本題為互動題**，你要透過評分系統提供的函數，來模擬空、白跟伊綱玩遊戲的過程。請引用標頭檔 Blackjack.h ，並且完成 void play() 這個函數。請使用以下的函數們與評分系統溝通，並且 **請勿進行任何輸入（stdin）、輸出（stdout）**（輸出到 stderr 將被忽略）。

1. `void play()` ：這是你要寫出的函數。每當評分系統要進行一次新的牌局時，會呼叫 `play()` 函數。你必須在這個函數裡面，與評分系統互動，來完成「21 點」的遊戲。當你的籌碼已經超過 1.3×10^6 時，評分系統會停止呼叫 `play()` 函數，並且判定你勝利。獲得 AC～

2. `vector<int> init_game(int bet)` ：在 `void play()` 中，你必須需要這個函數，來進行下注。傳入的 `int bet` 是你在此牌局中想要下注的金額，**此金額必須是一個正整數，並且不能超過你現在手中的資金。**回傳值會是一個長度為 3 的 `vector<int>` ，前兩個元素是閒家手上的兩張牌，第三個元素是莊家的明牌。撲克牌表達的方式為：$A, 2, 3, \ldots, 9, 10, J, Q, K$ 分別以整數 $1, 2, 3, \ldots, 9, 10, 11, 12, 13$ 表示。

3. `int buy_insurance(int insurance_bet)` ：如果莊家的明牌是 A ，你 **必須呼叫此函數**。你必須傳入 `insurance_bet` ，代表你要買保險的金額，此金額不能超過 $\frac{B}{2}$。接著，這個函數如果回傳 1 ，代表莊家的牌真的是 blackjack ，這時請立刻呼叫 `stand()` ，來結束這個牌局，否則牌局繼續。

4. `int double_down()` ：如果你想要進行 double down ，請呼叫此函數。函數的回傳值為閒家得到的第三張牌。呼叫完此函數後，請立刻呼叫 `stand()` 。

5. `vector<int> split()` ：如果你想要進行 split ，請呼叫此函數。函數的回傳值是一個長度為 2 的 `vector<int>` ，第一個元素是第一個牌堆得到的牌，第二個元素是第二個牌堆得到的牌。

6. `int hit_1()` ：**在你呼叫完 split 後**，你必須對第一堆牌堆決定要不要加牌。如果你想對第一堆牌堆加牌，請呼叫此函數。函數的回傳值，是你從莊家那邊得到的牌。

7. `void stand_1()` ：**在你呼叫完 split 後**，你必須決定第一堆牌什麼時候才要停止加牌。如果你打算在第一堆牌堆停止加牌，**或者是第一牌堆已經爆牌（busted）時**，請呼叫此函數。

8. `int hit_2()` ：**在你呼叫完 stand_1() 後**，你必須對第二牌堆決定要不要加牌。如果你想對第二堆牌堆加牌，請呼叫此函數。函數的回傳值，是你從莊家那邊得到的牌。

9. `vector<int> stand_2()`：**在你呼叫完 stand_1() 後**，你必須決定第二堆牌什麼時候要停止加牌。如果你打算在第二牌堆停止加牌，**或者是第二牌堆已經爆牌（busted）時**，請呼叫此函數。函數的回傳值是莊家最後的牌。之後請你結束 `play()` 函數，評分系統會結算你在這輪得到的金額。

10. `int hit()` ：如果你打算加一張牌，請呼叫此函數。函數的回傳值就是你從莊家那裡得到的牌。

11. `vector<int> stand()` ：如果你打算停止加牌，**或者是已經爆牌**時，請呼叫此函數。此函數的回傳值是莊家最後的牌。之後請你結束 `play()` 函數，評分系統會結算你在這輪得到的金額。

除此之外，評分系統還有一些實用的函數，分別如下：

1. `bool is_blackjack(vector<int> v)` ：這個函數是判斷手牌是不是 blackjack 使用。你必須把想要判斷的手牌存到 `vector<int> v` 裡面，之後把這個東西傳到這個函數裡面。如果手牌是 blackjack ，此函數會回傳 `true` ，否則會回傳 `false` 。

2. `vector<int> cal_point(vector<int> v)` ：這個函數是計算手牌 `vector<int> v` 的點數。回傳值是一個 `vector<int>` ，裡面會 **有小到大**儲存所有可能的點數。**如果已經爆牌了，此函數會回傳一個只有包含 -1 的 vector<int>**。

3. `int cal_max_point(vector<int> v)` ：這個函數是計算手牌 `vector<int> v` 中，**最大的可能點數（沒有爆牌的情況下）**，並且把這個點數回傳。**如果已經爆牌了，此函數會回傳 -1**。

4. `int win(vector<int> a, vector<int> b)` ：此函數必須傳入 **閒家手牌** `vector<int> a`，**莊家手牌** `vector<int> b` ，此函數會判斷閒家跟莊家的輸贏狀態。如果是平手，會回傳 0 ，如果閒家贏了，會回傳 1 ，否則如果莊家贏了，會回傳 -1 。

5. `int bet_you_have()` ：此函數會回傳你當前有的籌碼數量是多少。

Sample Code

以下程式碼是一個簡單的範例，可以在 CMS 的附件中找到～

```
#include "Blackjack.h"

void play() {
    int now_bet = bet_you_have();
    vector<int> v = init_game(now_bet / 10);
    vector<int> aa;
    aa.push_back(v[0]); aa.push_back(v[1]);
    if (v[2] == 1) {
        int x = buy_insurance(now_bet / 30);
        if (x == 1) {
            vector<int> xx = stand();
            // do something
            return;
        }
```

```
15      }
16      if (is_blackjack(aa)) {
17          // you are blackjack
18          vector<int> xx = stand();
19          // do something
20          return;
21      }
22      if (v[0] == v[1]) {
23          // split
24          vector<int> vv = split();
25          vector<int> c1, c2;
26          c1.push_back(aa[0]); c1.push_back(vv[0]);
27          c2.push_back(aa[1]); c2.push_back(vv[1]);
28          while (cal_max_point(c1) != -1
29              && cal_max_point(c1) <= 14) {
30              // hit until point > 14
31              int x = hit_1();
32              c1.push_back(x);
33          }
34          stand_1();
35          vector<int> xx = stand_2();
36          // do something
37          return;
38      }
39      if (cal_max_point(aa) <= 10) {
40          double_down();
41          stand();
42          // do something
43          return;
44      }
45      hit();
46      stand();
47      // do something
48  }
```

Sample

Request	Response
-- init_game(100) -- hit() -- stand() -- -- init_game(100) -- buy_insurance(0) -- stand() -- -- init_game(500000) -- split() -- stand1() hit2() -- stand2() -- -- init_game(500000) -- double_down() -- stand() --	play() -- {1, 9, 5} -- 1 -- {5, 10, 12} play() -- {3, 4, 1} -- 1 -- {1, 13 } play() -- {5, 5, 2} -- {1, 10} -- -- 7 -- {2, 13, 12} play() -- {1, 3, 10} -- 10 -- {10, 2, 1, 1, 3}

Local testing tool

範例評分程式會按照以下的格式讀取輸入：

- 第一列：一個正整數 $seed$ ，代表亂數的種子。

請注意，使用自己的測試資料進行測試時，你會得到一些訊息，訊息分別如下：

- Accepted：你通過了伊綱的考驗。
- Wrong_answer：你失敗了，或者是在過程中有出現 **不合法的操作**，詳細的內容可以參考後面的錯誤訊息。

在 CMS 的附件中，我們有附上 Blackjack.cpp 這個檔案，內部已經有寫好一些東西的檔案。直接上傳該檔案並不會導致 Compile Error ，並且往後你要寫的程式都可以參考這份檔案，**並在這份檔案內進行修改、編譯以及執行**。

請注意，檔案內有兩行註解分別為「do not modify above」和「do not modify below」，這兩行意外著希望你盡量只更動被這兩行夾住的區域，若你對互動題並沒有很熟悉，請不要更動外面的區域！

Hint 1

十條盟約：

1. 這個世界禁止一切殺傷、戰爭與掠奪。
2. 所有的糾紛一律以遊戲勝負解決。
3. 遊戲需賭上雙方判斷對等的賭注。
4. 在不違反 3. 的情況下，遊戲內容、賭注皆不限制。
5. 受挑戰方有權決定遊戲的內容。
6. 舉凡〈向盟約宣誓〉（盟約に誓って）的打賭絕對要遵守。
7. 集團間的糾紛應指定全權代理人。
8. 遊戲中若有不正當行為，一旦敗露即視同敗北。
9. 以神之名宣布，以上各條皆為絕對不變的規則。
10. 大家一起和平地玩遊戲吧 !!

Hint 2

空、白發現要戰勝負期望值的遊戲好難，因此他們希望你趕快得到 AC ！

Hint 3

如果你的程式通過的機率是 0.5 ，那你期望上傳 $\frac{1}{(0.5)^5} = 32$ 次，就會 AC 囉～

配分

在一個子任務的「測試資料範圍」的敘述中，如果存在沒有提到範圍的變數，則此變數的範圍為 Input 所描述的範圍。

子任務編號	子任務配分	測試資料範圍
1	100%	無特殊限制

注意！測試資料只有五筆！