Problem F. Thunder Bluff

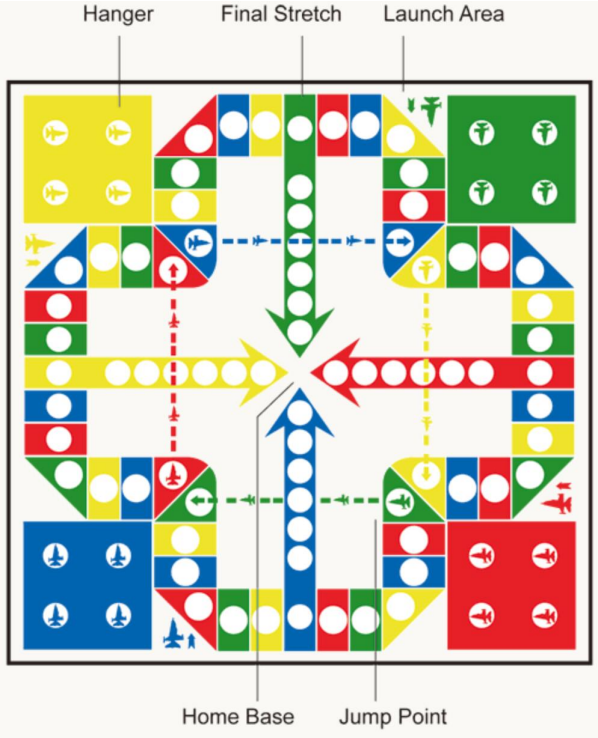
Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

–*"Thunder Bluff is a beautiful and peaceful pla...."*

–*"I know! Now,let's play Airplane chess!"*

–*"What?"*

The board for the game is given in the picture below:



The game consists of four players, with each controlling one set of 4 pieces in the color of yellow, green, red and blue representing the airplanes. Initially, four pieces are placed facing up in the *Hanger* area of the respective color.

Starting with yellow, the players take turns to roll a dice in the order yellow, green, red and blue and. The rules are as follows(The *Hanger*/*Launch Area*/*Final Stretch*/*Jump Point*/*Home base*/ refer to the picture markings):

1. **Action rule:** Player can move one piece outside *Hanger* for x steps if the dice faced up with number x, or move one piece from *Hanger* to *Launch* if $x = 6$. If all four pieces are in *Hanger* and $x \neq 6$, the round is skipped.Starting from the *Launch* area, pieces move **clockwise** around the board, until reaching *Final Stretch* where the **Final stretch rule** is applied.

2. **Final stretch rule:** When a player's piece has reached the *Final Stretch* of its own color the piece, the direction will change towards its home base. If one piece will land exactly on the innermost grid of the *Final Stretch*, it will return *Hanger* facing down as the mark of completion. If the dice roll value is too large to land exactly on the innermost grid, it must continue the remaining steps backward along the final stretch.

3. **Stopping time:** The first player to have all four pieces facing down in the *Hanger* wins and the game ends.

4. **Pity rule:** In order to spice the game up, the pity rule is applied: if one skipped for consecutive 10

rounds, one piece will move into the launch area after skipping the 10-th round. Note that this piece in *Launch* can always move in the next round, and therefore the skipping streak will always be terminated by the pity rule, which means after *Launch*, the streak will become 0.

5. **Bonus round rule:** The player rolled a 6 will receive a *bonus turn*. After taking an action according to the Action rule, it will play an extra *bonus turn*. A player can receive another *bonus turn* in the previous *bonus turn* – with good enough luck, one can play infinite bonus turns until the game ends.

6. **Combo rule:** When one piece is going to land on a grid of its own color outside the *Final Stretch*, a *combo* will occur and move the piece four steps forward, and land on the next grid of the same color. The *combo* can not cause another *combo* in one round.

7. **Shortcut rule:** When one piece is going to land on a jump point of its own color, the player jumps by following the *shortcut* (indicated by the dotted line) to the grid across the board. Note that only one *combo* will happen in one *shortcut* journey: the piece will stay at the end of the *shortcut* if one *combo* happens before the *shortcut*, otherwise a *combo* will occur after the *shortcut*.

8. **Stacking rule:** After the settlement of *combo* and *shortcut*, one piece lands on a grid outside *Launch* area that already has some pieces of the same color, then those pieces will *stack* together. Pieces that are stacked will then **move together as one unit**.

9. **Proctect rule:** A piece is **protected** on the *Final Stretch* of its own color (including the outermost grid).

10. **Battle rule:** After the settlement of *combo* and *shortcut*, one piece lands on a grid with some opponent's pieces after one move, a battle occurs unless one side is **protected**. When a pieces of player A(ttacker) are arriving on this grid and d pieces of player D(efender) is on this grid originally, then $\min(a, d-1)$ pieces of player A and $\min(a, d)$ pieces of player D will be shot down and sent back to there respective hangers facing up.

In addition, a scoring system is included in this game according to the following rules:

1. Initially each player has 0 points. 2. When a player's piece reaches its home base, it receives $5x$ points (x is the number of pieces of other players that **haven't reached** the home base). 3. When some pieces of player A and B have a fight, where A is the **attacker** and B is the **defender**, if a pieces of player A and b pieces of player B are shot down and sent back to the hangers, then A receives $5b$ points and B receives $2a$ points. 4. For each player, if it rolls the number x for a consecutive y times, it receives $(6-x)*(y-1)^2$ points. Here only the **maximal consecutive identical rolls** are rewarded with points. Formally, suppose a player rolls x on its $i^{th}, (i+1)^{th}...(i+k)^{th}$ roll, then it receives $(6-x)*k^2$ points if and only if : 1. $i=1$ or the $(i-1)^{th}$ roll is not x 2. the $(i+k+1)^{th}$ roll does not exist (the game ends before it) or the $(i+k+1)^{th}$ roll is not x. 5. The winner receives an additional 50 points.

Now Alice, Bob, Carol and David are playing this game. Their colors are yellow, green, red and blue respectively. The players have one thing in common: if they roll 6 and have pieces facing up in the hanger, they will choose to launch a piece instead of moving the pieces out of the hanger. Beside that, if there are multiple choices of moves, each player has its own strategies:

Alice will always choose the move that **maximizes the number of stacked pieces** (that is, maximize the number of pieces belonging to Alice on the starting grid and landing grid, disregarding the possible fights after the move). If there are still multiple choices, Alice will choose to move the piece(s) closest to the home base.

Bob is aggressive and eager to have a fight with other players, thus he will always choose the move that gets **the maximum score** by shooting down other players' pieces. If there are still multiple choices, Bob will choose to move the piece(s) closest to the home base.

Carol wants his pieces to travel as fast as possible, so he will always choose the move that **maximizes the distance of grids** between the starting grid and the landing grid, which can be **negative or zero** in some special cases. If there are still multiple choices, Carol will choose to move the piece(s) closest to the home base.

David doesn't want any complex strategies and will always choose to move the piece(s) **closest to the**

launch area(that is, the maximum distance between the current grid and the home base).

Here we define **closest to the home base** as having the minimum number of grids between the current grid and the respective home base, disregarding jumps.

Now you are given the result of each dice roll, you should predict the winner of the game and the final score of each player.

In order to avoid huge input, you are given four secret parameters x, y, z, w and the result of each dice roll is generated by the following code:

```
uint32_t x, y, z, w;
uint32_t Xor128() {
        uint32_t t;
        t = x ^ (x << 11);
        x = y; y = z; z = w;
        return w = w ^ (w >> 19) ^ (t ^ (t >> 8));

}
int GetDice() {
        return Xor128() % 6 + 1;
}
```

CAUTION: There is some difference between the rules described in this problem and the rules of the Airplane Chess in the human world, so please read the rules above carefully.

Input

The input consists of multiple test cases.

The first line contains one integer T ($1 \leq T \leq 50$) denoting the number of test cases.

Each test case consists of one line containing four integers x, y, z, w ($0 \leq x, y, z, w \leq 2^{32} - 1$).

It's guaranteed that according to the rule and data, the game will ends in finite rounds.

Output

For each test case, output five lines:

The first line displays the winner, in the form "NAME win!". ("NAME"is replace by "Alice "Bob "Carol "David").

The next four lines display the final score of each player, in the form "NAME: score sorted by the score from large to small. If there are two players whose score are equal, output the player whose name has smaller lexicographical order first.

See the Sample Output for better understanding.

Example

standard input	standard output
1 1 1 1 1	Bob win! Bob: 269 Alice: 213 Carol: 118 David: 99