```
/*
COVACI ANDREI GABRIEL
C.N. "Emanuil Gojdu" Oradea
ACSL - Contest 4
Junior Division
*/


#include <iostream>
#include <fstream>

using namespace std;

#define LENGTH 52

ifstream in("input.in");
ofstream out("output.out");

int board[53][3];
int adversaryPos[4], pos, nrRolls, rolls[200];

void setPart(int start, int end, int id){
   for(int i = start; i <= end; i++){
      board[i][0]= id;
   }
}

void setSquares(int start, int end){
   int counter = 0;

   for(int i = start; i <= end; i++){
      if(counter == 4){
         setPart(i, i, 0);
         counter = 0;
      }else{
         setPart(i, i, 1);
         counter++;
      }
   }
}

void setUpPrimes(){
   for(int i = 1; i < LENGTH; i++){
      board[i][1] = 1;
```

```
    }

    board[1][1] = 0;
    board[2][1] = 1;

    for(int i = 2; i * i <= LENGTH; i++){
        if(board[i][1] == 1){
            for(int j = 2 * i; j <= LENGTH; j += i){
                board[j][1] = 0;
            }
        }
    }
}

void setUpPerfectSquares(){
    for(int i = 3; i * i <= LENGTH; i++){
        board[i * i][2] = 1;
    }
}

void setUpBoard(){
    setPart(1, 3, 0);
    setSquares(4, 22);
    setPart(23, 26, 0);
    setPart(27, 27, 1);
    setPart(28, 31, 0);
    setSquares(32, 50);
    setPart(51, 52, 0);
}

bool isOccupied(int pos){
    if(pos == adversaryPos[1] || pos == adversaryPos[2] || pos == adversaryPos[3]){
        return true;
    }
    return false;
}

int main(int argc, char const *argv[])
{
    setUpBoard();
    setUpPrimes();
    setUpPerfectSquares();

    for(int o = 0; o < 5; o++){
```

```
in >> adversaryPos[1] >> adversaryPos[2] >> adversaryPos[3] >> pos >> nrRolls;
for(int i = 1; i <= nrRolls; i++){
   in >> rolls[i];
}

bool gameover = false;
int oldPos = pos, newPos;
for (int i = 1; i <= nrRolls; i++){
   newPos = oldPos + rolls[i];

   if(newPos == 52){
      gameover = true;
      break;
   }
   if(newPos > 52){
      newPos = oldPos;
   }else{
      if(isOccupied(newPos)){
         newPos = oldPos;
      }else{
         if(board[newPos][1] == 1){
            int j;
            for(j = 1; j <= 6; j++){
               if(isOccupied(newPos + j)){
                  break;
               }
            }
            if(newPos + j - 1 > 52){
               newPos = newPos;
            }else{
               newPos += j - 1;
            }
            if(newPos == 52){
               gameover = true;
               break;
            }
         }else if(board[newPos][2] == 1){
            int j;
            for(j = 1; j <= 6; j++){
               if(isOccupied(newPos - j)){
                  break;
               }
            }
            newPos -= j - 1;
```

```
            }else{
               bool horThenVer = false;
               for(int j = oldPos + 1; j < newPos; j++){
                  if(board[j][0] == 1 && board[j + 1][0] == 0){
                     horThenVer = true;
                     break;
                  }
               }
               if(horThenVer){
                  bool foundPos = false;
                  for(int j = newPos; j >= oldPos; j--){
                     if(j % rolls[i] == 0 && !isOccupied(j)){
                        newPos = j;
                        foundPos = true;
                        break;
                     }
                  }
                  if(!foundPos){
                     newPos = oldPos;
                  }
                  if(foundPos && newPos == 52){
                     gameover = true;
                     break;
                  }
               }
            }
            oldPos = newPos;
         }
      }
   }
   if(gameover){
      out << "GAME OVER";
   }else{
      out << oldPos;
   }
   out << endl;
  }

  return 0;
}
```