1 Beginning — Clock

Write a program that computes the total number of seconds in a given time period specified by hours, minutes, and seconds. The input of your program will consist of three nonnegative integers, \( h \), \( m \), and \( s \), representing hours, minutes, and seconds, respectively. The output should be the total number of seconds in this time period.

Example:

Enter hours: 23
Enter minutes: 59
Enter seconds: 59
Total seconds: 86399
Write a program to compute the yearly cost of a car based upon the following information:

- the purchase price (in dollars) of the car;
- the expected lifetime (in miles) of the car;
- the expected number of miles per year the car will be driven;
- the estimated gas mileage of the car in miles per gallon; and
- the expected average price of gasoline (in dollars per gallon) over the lifetime of the car.

The total cost per year should include the purchase price, averaged over the expected lifetime of the car, plus the expected cost of the gasoline used per year. You should assume that the five input items above are positive floating-point values.

Example:

Enter purchase price: 10000
Enter lifetime: 100000
Enter miles per year: 20000
Enter gas mileage: 20
Enter gas price: 2.5
Yearly cost = $4500.0
Your program must accept orders from the menu and print out the total cost including 7% tax. The items are ordered by number. A -1 will be used to identify the end of the order.

<table>
<thead>
<tr>
<th>Item number</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7.99</td>
</tr>
<tr>
<td>2</td>
<td>6.99</td>
</tr>
<tr>
<td>3</td>
<td>2.99</td>
</tr>
<tr>
<td>4</td>
<td>.99</td>
</tr>
</tbody>
</table>

Example 1:
Input: 1
2
3
-1

Output: 11.05
In the guessing game, the player thinks of a number between 1 and 32 (including 1 and 32). The computer program tries to guess the number within 6 guesses. The player will answer whether the guess is correct or whether the computer is too high or too low on the guess.

Example 1:

The computer guesses 16.
Is this correct? (Y or N): N
Is it too high (H) or too low (L): L

The computer guesses 24.
Is this correct? (Y or N): N
Is it too high (H) or too low (L): L

The computer guesses 28.
Is this correct? (Y or N): N
Is it too high (H) or too low (L): H

The computer guesses 26.
Is this correct? (Y or N): N
Is it too high (H) or too low (L): H

The computer guesses 25.
Is this correct? (Y or N): Y
A palindrome is a string of letters that is same front to back or back to front. That is, it is symmetrical about the middle of the string. The string, aabbba, is a palindrome. The string, aabbcc, is not. Your program must accept strings up to 20 chars in length. The input can be one char per line terminated by a period that is not counted as part of the string.

Example 1
Input:
a a b b b a a
Output: palindrome

Example 2
Input:
a a a b b a a
Output: not a palindrome
6 Beginning — Making Change

Write a program that takes a nonnegative integer monetary value as input and produces as output the number of each denomination of coin that is needed to achieve that value. The values of the coin denominations to be used are 1, 5, 10, 25, 100, 500, 1000, and 2000. You must give preference to the higher denominations, so that a minimum number of coins is used; thus, even though there are several ways to achieve a value of 24, we must use as many 10s as we can (2), then as many 5s as we can (0), then as many 1s as we can (4).

Example 1:

Enter value: 24
2000: 0
1000: 0
500: 0
100: 0
25: 0
10: 2
5: 0
1: 4

Example 2:

Enter value: 2199
2000: 1
1000: 0
500: 0
100: 1
25: 3
10: 2
5: 0
1: 4
1 Advanced — Elapsed Time

Write a program that takes as input a starting time and an ending time and produces as output the elapsed time. The inputs should be presented as hours, minutes, and seconds, using a 24-hour clock. Thus, the hours are nonnegative integers less than 23, and the minutes and seconds are nonnegative integers less than 60. You may assume that the times occur on the same day, and that the ending time is no earlier than the starting time.

Example:

Enter starting hour: 9
Enter starting minute: 30
Enter starting second: 45
Enter ending hour: 15
Enter ending minute: 30
Enter ending second: 0
Elapsed time:
  5 hours
  59 minutes
  15 seconds
2 Advanced — Car Costs

Write a program to determine the least expensive of four given cars based on the following information:

- the expected number of miles per year that the car will be driven;
- the expected average price of gasoline (in dollars per gallon) over the lifetime of the car; and
- for each car:
  - the purchase price in dollars;
  - the expected lifetime of the car in miles; and
  - the estimated gas mileage in miles per gallon.

You should assume that each of the above inputs is a positive floating-point number. Your program should identify which car has the lowest annual cost; in addition, it should report the annual cost for that car. The annual cost should include the purchase price, averaged over the expected lifetime of the car, and the expected cost of gasoline per year for that car.

Example:

Enter miles per year: 15000
Enter gas price: 2.599
Car 0:
  Enter purchase price: 10000
  Enter lifetime: 100000
  Enter gas mileage: 25
Car 1:
  Enter purchase price: 15000
  Enter lifetime: 120000
  Enter gas mileage: 21.7
Car 2:
  Enter purchase price: 20000
  Enter lifetime: 150000
  Enter gas mileage: 20
Car 3:
  Enter purchase price: 25000
  Enter lifetime: 200000
  Enter gas mileage: 30
Least expensive car: 0
Cost per year: $3059.4
A3 Coin Game

In the coin game, a coin is passed x number of steps. The first step is to player 1. When the coin stops, the player who has the coin is out of the game. The coin is then passed x more steps among the players that are left. The winner is the last player still in the game. Write a program that accepts the number of players (1-100) and the number of steps (1-100). Print out the each player's number as each is eliminated. Print out the number of the winning player.

Example 1:

Enter number of players: 5
Enter number of steps: 3

Output is:

Player 3 is out!
Player 1 is out!
Player 5 is out!
Player 2 is out!
Player 4 is the winner!
In the guessing game, the player thinks of a number between 1 and N (including 1 and N). The computer program tries to guess the number within \(\log_2 N\) guesses. For N=32, this is 5 guesses, for N=1024, it is 10 guesses.

The player will answer whether the guess is correct or whether the computer is too high or too low on the guess.

Example 1:

Enter N: 32

The computer guesses 16.
Is this correct? (Y or N): N
Is it too high (H) or too low (L): L

The computer guesses 24
Is this correct? (Y or N): N
Is it too high (H) or too low (L): L

The computer guesses 28
Is this correct? (Y or N): N
Is it too high (H) or too low (L): H

The computer guesses 26
Is this correct? (Y or N): N
Is it too high (H) or too low (L): H

The computer guesses 25
Is this correct? (Y or N): Y
5 Advanced — Root Finder

Write a program to find a root of a polynomial of the form:

\[ p(x) = x^5 + ax^4 + bx^3 + cx^2 + dx + e \]

where \( a, b, c, d, \) and \( e \) are given floating-point numbers. A polynomial of this form has a graph resembling the one shown below; in particular, as \( x \) approaches \(-\infty\), \( p(x) \) will approach \(-\infty\), and as \( x \) approaches \(\infty\), \( p(x) \) will approach \(\infty\). Because \( p(x) \) must go from negative to positive at least once as \( x \) increases, it will always cross the \( x \)-axis at least once. Each value of \( x \) for which \( p(x) \) reaches the \( x \)-axis (i.e., \( p(x) = 0 \)) is called a root of \( p \). Your program must find a floating-point approximation to one real root of \( p \). Specifically, if your program produces a value of \( \hat{x} \), then there must be a real root \( x \) such that \( |x - \hat{x}| \leq 0.001 \).

![Graph of the polynomial](image)

Example: [The polynomial in this example has only one real root.]

Enter coefficient for \( x^4 \): -2.7
Enter coefficient for \( x^3 \): 0
Enter coefficient for \( x^2 \): 0
Enter coefficient for \( x^1 \): 0
Enter coefficient for \( x^0 \): -1.6
Root = 2.72900390625
In our magic 3 by 3 squares, each number between 1 and 9 is used just once. The sum of each column, each row and each diagonal is the same, 15. Below is an example of a magic square. We will also number the cells with letters. The input will be a letter designating a cell and an integer between 1 and 9. The output will be a magic square with that number in that cell, each number between 1 and 9 used only once and the sum of each row, column and diagonal being 15. If such a square is not possible, then that message will be displayed.

```
8 3 4
1 5 9
6 7 2
```

Sample Magic Square

```
a b c
 d e f
g h i
```

Labeling of cells

Example 1:
Input: b 7
Output:
2 7 6
9 5 1
4 3 8

Example 2:
Input: e 7
Output: no possible magic square
Test Case 1:

Enter starting hour: 10
Enter starting minute: 59
Enter starting second: 59
Enter ending hour: 11
Enter ending minute: 0
Enter ending second: 0
Elapsed time: 
  0 hours
  0 minutes
  1 seconds

Test Case 2:

Enter starting hour: 0
Enter starting minute: 0
Enter starting second: 0
Enter ending hour: 23
Enter ending minute: 59
Enter ending second: 59
Elapsed time: 
  23 hours
  59 minutes
  59 seconds

Retest: Do the above tests, plus the following:

Test Case 3:

Enter starting hour: 13
Enter starting minute: 23
Enter starting second: 34
Enter ending hour: 15
Enter ending minute: 20
Enter ending second: 10
Elapsed time: 
  1 hours
  56 minutes
  36 seconds
Note: Because the calculations are floating-point, the results may not match exactly; however, they should be pretty close.

Test Case 1:

Enter miles per year: 15432.73
Enter gas price: 2.678
Car 0:
   Enter purchase price: 14999.99
   Enter lifetime: 12345.67
   Enter gas mileage: 27.8
Car 1:
   Enter purchase price: 22222.22
   Enter lifetime: 150000
   Enter gas mileage: 33.3
Car 2:
   Enter purchase price: 550
   Enter lifetime: 10000
   Enter gas mileage: 11.1
Car 3:
   Enter purchase price: 40000
   Enter lifetime: 200000
   Enter gas mileage: 20
Least expensive car: 1
Cost per year: $3527.436776572168

Retest: The above plus the following:

Test Case 2:

Enter miles per year: 25000
Enter gas price: 2.999
Car 0:
   Enter purchase price: 11111.11
   Enter lifetime: 11111.1
   Enter gas mileage: 22.22
Car 1:
   Enter purchase price: 1234.5
   Enter lifetime: 54321
   Enter gas mileage: 19
Car 2:
   Enter purchase price: 22222.22
   Enter lifetime: 88888
   Enter gas mileage: 52.3
Car 3:
   Enter purchase price: 4321.12
   Enter lifetime: 55555.5
   Enter gas mileage: 32.1
Least expensive car: 3
Cost per year: $4280.175726437409
A3

Coin Game Tests

Print out the each player’s number as each is eliminated.
Print out the number of the winning player.

Test 1:
Enter number of players: 5
Enter number of steps: 3

Output is:
Player 3 is out!
Player 1 is out!
Player 5 is out!
Player 2 is out!
Player 4 is the winner!

Test 2: number of players 1 number of steps is 4

Output:
Player 1 is the winner!

Test 3: number of players 10 number of steps is 12

Output:
Players eliminated - 2,5,9,6,4,8,7,3,10 winner is 1

Second set of tests – all of the above and

Test 4: number of players 3 step is 1
Players eliminated – 1, 2 winner is 3
Tests for Guessing Game

Test 1: (the computer guesses don’t have to be as shown, but if it is not, it probably doesn’t work – but you will have to respond appropriately for the intended answer.).

It is also incorrect if it does not answer correctly in \(\log_2 N\) guesses

Input 32 for max number
The computer guesses 16.
Is this correct? (Y or N): N
Is it too high (H) or too low (L): L

The computer guesses 24
Is this correct? (Y or N): N
Is it too high (H) or too low (L): L

The computer guesses 28
Is this correct? (Y or N): N
Is it too high (H) or too low (L): H

The computer guesses 26
Is this correct? (Y or N): N
Is it too high (H) or too low (L): H

The computer guesses 25
Is this correct? (Y or N): Y

Test 2: (I won’t show the computer output or the N response – just answer L for too low

\(N = 32\)
Responses : L L L L L

Output: 32

Test 3:
\(N = 100\)
Responses: H H H H H (a max of 10 responses)
Output: 1

Retesting – just repeat the above tests plus the following

Input 50
Responses: H H H H H (max 6)
Output: 50
Note: All results should be within ±0.001 of the shown root.

Test 1:

Enter coefficient for $x^4$: 2.5  
Enter coefficient for $x^3$: 2.5  
Enter coefficient for $x^2$: 1.25  
Enter coefficient for $x^1$: 0.3125  
Enter coefficient for $x^0$: 0.03125  
Root = $-0.5$

Test 2:

Enter coefficient for $x^4$: -9  
Enter coefficient for $x^3$: 26  
Enter coefficient for $x^2$: -34  
Enter coefficient for $x^1$: 21  
Enter coefficient for $x^0$: -5.00001  
Root = 5.0

Retest: Use all of the above tests, plus:

Test 3:

Enter coefficient for $x^4$: -5  
Enter coefficient for $x^3$: 10  
Enter coefficient for $x^2$: -10  
Enter coefficient for $x^1$: 5  
Enter coefficient for $x^0$: -1  
Root = 1.0
Test 1:
Input: b 7
Output:
\[
\begin{array}{ccc}
2 & 7 & 6 \\
9 & 5 & 1 \\
4 & 3 & 8 \\
\end{array}
\]

Test 2:
Input: c 7
Output: no possible magic square

Test 3:
Input: g 4
Output:
\[
\begin{array}{ccc}
2 & 7 & 6 \\
9 & 5 & 1 \\
4 & 3 & 8 \\
\end{array}
\]

Test 4:
Input: f 7
Output:
\[
\begin{array}{ccc}
4 & 9 & 2 \\
3 & 5 & 7 \\
8 & 1 & 6 \\
\end{array}
\]
Retesting: all of the above plus

Test 5:
Input: f 9
Output:
\[
\begin{array}{ccc}
8 & 3 & 4 \\
1 & 5 & 9 \\
6 & 7 & 2 \\
\end{array}
\]

Test 6:
Input: f 8
Output: no possible magic square
Test Case 1:

Enter hours: 1
Enter minutes: 2
Enter seconds: 3
Total seconds: 3723

Test Case 2:

Enter hours: 0
Enter minutes: 0
Enter seconds: 0
Total seconds: 0

Retest: Do the above tests, plus the following:

Test Case 3:

Enter hours: 6
Enter minutes: 7
Enter seconds: 8
Total seconds: 22028
Test Cases

Note: Because the calculations are floating-point, the results may not match exactly; however, they should be pretty close.

Test Case 1:

Enter purchase price: 21562.73
Enter lifetime: 132463.7
Enter miles per year: 15732.87
Enter gas mileage: 23.38
Enter gas price: 2.779
Yearly cost = $4431.075620801704

Test Case 2:

Enter purchase price: 525.75
Enter lifetime: 9237.3
Enter miles per year: 12624.9
Enter gas mileage: 11.4
Enter gas price: 2.52
Yearly cost = $3509.325948850399

Retest: Do the above tests, plus:

Test Case 3:

Enter purchase price: 7000
Enter lifetime: 50000
Enter miles per year: 15000
Enter gas mileage: 24.5
Enter gas price: 2.75
Yearly cost = $3783.673469387755
Test 1:
Input: 1
  2
  3
-1

Output: 11.05

Test 2:
  2
  3
  1
  1
  1
  4
-1

Output: 28.84 (or 28.8365 okay)

Additional testing after one failed attempt

All of the above plus

Test 3:
  1
  1
-1

Output: 17.10 (or 17.0986)
Test 1: (the computer guesses don't have to be as shown, but if it is not, it probably doesn't work - but you will have to respond appropriately for the intended answer.).

It is also incorrect if it does not answer correctly in 5 guesses

The computer guesses 16.
Is this correct? (Y or N): N
Is it too high (H) or too low (L): L

The computer guesses 24
Is this correct? (Y or N): N
Is it too high (H) or too low (L): L

The computer guesses 28
Is this correct? (Y or N): N
Is it too high (H) or too low (L): H

The computer guesses 26
Is this correct? (Y or N): N
Is it too high (H) or too low (L): H

The computer guesses 25
Is this correct? (Y or N): Y

Test 2: ( I won't show the computer output or the N response - just answer L for too low Responses: L L L L L
Output: 32

Test 3: Responses: H H H H H
Output: 1

Retesting - just repeat the above tests plus Responses: H L L L L
Output: 15
Test 1
Input:
  a
  b
  b
  b
  a
  a
  .
Output: palindrome

Test 2
Input:
  a
  a
  a
  b
  b
  a
  a
  a
  .
Output: not a palindrome

Test 3
Input: aabbaa. (can be one per line)
Output: palindrome

Test 4
Input: a.
Output: palindrome

Retesting – all of the above plus
Input: abcdefgfedcba.
Output: palindrome
Test 1:

Enter value: 512
2000: 0
1000: 0
500: 1
100: 0
25: 0
10: 1
5: 0
1: 2

Test 2:

Enter value: 496
2000: 0
1000: 0
500: 0
100: 4
25: 3
10: 2
5: 0
1: 1

Test 3:

Enter value: 0
2000: 0
1000: 0
500: 0
100: 0
25: 0
10: 0
5: 0
1: 0

Retest:

Use the above tests, plus the following:

Test 4:

Enter value: 8128
2000: 4
1000: 0
500: 0
100: 1
25: 1
10: 0
5: 0
1: 3
// Beginning - Clock

import java.io.*;

public class ClockB {

    public static void main(String args[]) throws Exception {
        BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
        System.out.print("Enter hours: ");
        int hrs = Integer.parseInt(in.readLine());
        System.out.print("Enter minutes: ");
        int min = Integer.parseInt(in.readLine());
        System.out.print("Enter seconds: ");
        int sec = Integer.parseInt(in.readLine());
        System.out.print("Total seconds: " + (3600*hrs + 60*min + sec));
    }
}
import java.io.*;

public class CarB {

    public static void main(String[] args) throws Exception {
        BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
        System.out.print("Enter purchase price: ");
        double price = Double.parseDouble(in.readLine());
        System.out.print("Enter lifetime: ");
        double life = Double.parseDouble(in.readLine());
        System.out.print("Enter miles per year: ");
        double mpy = Double.parseDouble(in.readLine());
        System.out.print("Enter gas mileage: ");
        double mpg = Double.parseDouble(in.readLine());
        System.out.print("Enter gas price: ");
        double gasPr = Double.parseDouble(in.readLine());
        double years = life/mpy;
        System.out.println("Yearly cost=$" + (price/years + gasPr/mpg*mpy));
    }
}
Solution B3

```
#include "stdafx.h"
#include <iostream>
using std::cin;
using std::cout;
using std::endl;

int _tmain(int argc, _TCHAR* argv[])
{
    float cost[9];
    char* meal[9];
    cost[0] = 0;
    cost[1] = 7.99;
    cost[3] = 2.99;
    float sum = 0.0;
    int n = 0;
    while (n > -1)
    {
        cout << "Enter number of menu choice or -1 to get bill";
        cin >> n;
        cout << endl << "Enter number of menu choice or -1 to get bill";
        cin >> n;
        sum = sum + cost[n];
    }
    cout << "Total bill is " << sum * 1.07;
    return 0;
}
```
```cpp
#include "stdafx.h"
#include <iostream>
using std::cin;
using std::cout;
using std::endl;

int main(int argc, TCHAR* argv[])
{
    int n;
    char answer;
    cout << "Enter range of guesses, zero to what?";
    cin >> n;
    int high, low, guess;
    high = n + 1;
    low = -1;
    while (high - low > 0) {
        guess = (high + low) / 2;
        cout << "My guess is " << guess;
        cout << " Is my guess correct? N or Y?";
        cin >> answer;
        if (answer == 'Y') {return 0;}
        if (answer == 'N') {
            cout << "Is my guess too low (L) or too high (H)?";
            cin >> answer;
            if (answer == 'H') {
                high = guess;
            }
            if (answer == 'L') {
                low = guess;
            }
        }
    }
    return 0;
}
```
# palindrome.cpp

```cpp
// palindrome.cpp : Defines the entry point for the console application.

#include "stdafx.h"
#include <iostream>

using std::cin;
using std::cout;
using std::endl;

int _tmain(int argc, _TCHAR* argv[])
{
    int n = 0;
    char s[20];
    char x;
    cin >> x;
    s[n++] = x;
    while (x != '.')
    {
        cin >> x;
        s[n++] = x;
        cout « "\n step " <<n<<" value "<<x;
    }
    n--; n--;
    cout « "nn " << n;
    char palindrome = 't';
    for (i=0;i<n/2;i++)
    {
        if (s[i] != s[n-i]) palindrome = 'f';
    }
    if (palindrome == 't') cout << "\nstring is a palindrome";
    if (palindrome == 'f') cout << "\nstring is not a palindrome";
    cin >> x;
    return 0;
}
```

// 6 Beginning - Making Change

import java.io.*;

public class CoinsB {

    public static void main(String args[]) throws Exception {
        int denom[] = new int[] {2000, 1000, 500, 100, 25, 10, 5, 1};
        BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
        System.out.print("Enter value: ");
        int v = Integer.parseInt(in.readLine());
        for (int i = 0; i < denom.length; i++) {
            int num = v/denom[i];
            System.out.println(denom[i] + " = " + num);
            v = v - num*denom[i];
        }
    }
}
import java.io.*;

public class ClockA {

    public static void main(String[] args) throws Exception {
        BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
        System.out.print("Enter starting hour: ");
        int stHour = Integer.parseInt(in.readLine());
        System.out.print("Enter starting minute: ");
        int stMin = Integer.parseInt(in.readLine());
        System.out.print("Enter starting second: ");
        int stSec = Integer.parseInt(in.readLine());
        System.out.print("Enter ending hour: ");
        int endHour = Integer.parseInt(in.readLine());
        System.out.print("Enter ending minute: ");
        int endMin = Integer.parseInt(in.readLine());
        System.out.print("Enter ending second: ");
        int endSec = Integer.parseInt(in.readLine());
        if (endSec < stSec) {
            endSec = endSec + 60;
            endMin = endMin - 1;
        }
        if (endMin < stMin) {
            endMin = endMin + 60;
            endHour = endHour - 1;
        }
        System.out.println("Elapsed time:");
        System.out.println(" 
+ (endHour - stHour) + " hours");
        System.out.println(" 
+ (endMin - stMin) + " minutes");
        System.out.println(" 
+ (endSec - stSec) + " seconds");
    }
}
// 2 Advanced - Car Costs

import java.io.*;

public class CarA {

  public static final int NUM = 4;

  public static void main(String[] args) throws Exception {
    BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
    System.out.print("Enter miles per year: ");
    double mpy = Double.parseDouble(in.readLine());
    System.out.print("Enter gas price: ");
    double gasPr = Double.parseDouble(in.readLine());
    double[] price = new double[NUM];
    double[] life = new double[NUM];
    double[] mpg = new double[NUM];
    double[] years = new double[NUM];
    double[] cpy = new double[NUM];
    double minCost = 0.0;
    int m = 0;
    for (int i = 0; i < NUM; i++) {
      System.out.println("Car" + i + ":");
      System.out.print("Enter purchase price: ");
      price[i] = Double.parseDouble(in.readLine());
      System.out.print("Enter lifetime: ");
      life[i] = Double.parseDouble(in.readLine());
      System.out.print("Enter gas mileage: ");
      mpg[i] = Double.parseDouble(in.readLine());
      years[i] = life[i]/mpy;
      cpy[i] = price[i]/years[i] + gasPr/mpg[i]*mpy;
      if (i == 0 || cpy[i] < minCost) {
        minCost = cpy[i];
        m = i;
      }
    }
    System.out.println("Least expensive car: "+m);
    System.out.println("Cost per year: "+minCost);
  }
}
Solution – Coin Game

#include <stdio.h>

int main()
{
    //Initialization
    int stepping = 0;
    int numberofplayers = 0;
    int count = 1;
    int tempcount = 1;
    int outs = 1;
    int i = 0;
    int x = 0;

    //User Input
    printf("Please enter the stepping: ");
    scanf("%d", &stepping);
    printf("Please enter the number of players: ");
    scanf("%d", &numberofplayers);

    //Declare array based on number of players
    char players[numberofplayers-1];

    //Assigning players to array
    for (x = 0; x < numberofplayers-1; x++)
    {
        players[x] = x;
    }

    //Runs through the coin game until 1 person is left
    while (outs != numberofplayers)
    {
        if (players[i] != -1)
        {
            if (count == stepping)
            {
                printf("Player %d is out! \n", i);
                outs++;
                count = 0;
                players[i] = -1;
            }
            count++;
        }
    }
}
/** guess.cpp : Defines the entry point for the console application. **/ 

#include "stdafx.h"
#include <iostream>

using std::cin;
using std::cout;
using std::endl;

int _tmain(int argc, _TCHAR* argv[])
{
    int n;
    char answer;
    cin<<"enter range of guesses, zero to what?";
    cin>> n;
    int high, low, guess;
    high = n+1;
    low = -1;
    while (high-low > 0){
        guess = (high+low)/2;
        cout<<"my guess is "«guess;
        cout<<"is my guess correct? N or Y";
        cin>>answer;
        if (answer == 'Y') return 0;
        if (answer == 'N')
            cout<<"Is my guess too low (L) or too high (H)?";
            cin>> answer;
            if (answer == 'H'){
                high = guess;
            }
            if (answer == 'L'){
                low = guess;
            }
    }
    return 0;
}
// 5 Advanced - Root Finding

import java.io.*;

public class Roots {
    static double[] coef = new double[5];

    public static void main(String[] args) throws Exception {
        BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
        for (int i = 4; i >= 0; i--) {
            System.out.print("Enter coefficient for x^" + i + ": ");
            coef[i] = Double.parseDouble(in.readLine());
        }
        double lo, hi;
        // We want the value at lo to be <= 0, and the value at hi to be >= 0
        if (eval(0.0) < 0.0) {
            lo = 0.0;
            hi = 1.0;
            while (eval(hi) < 0.0) {
                hi = hi*2.0;
            }
        } else {
            hi = 0.0;
            lo = -1.0;
            while (eval(lo) > 0.0) {
                lo = lo*2.0;
            }
        }
        // Use a binary search to find a root
        while (hi - lo > 0.0005) {
            double mid = (hi + lo)/2.0;
            if (eval(mid) > 0.0) {
                hi = mid;
            } else {
                lo = mid;
            }
        }
        System.out.println("Root= " + hi);
    }

    /* Evaluates the polynomial at x using Horner's Rule:
     */
    public static double eval(double x) {
        double y = 1.0;
        for (int i = 4; i >= 0; i--) {
            y = y*x + coef[i];
        }
        return y;
    }
}
#include "stdafx.h"
#include <iostream>
using std::cin;
using std::cout;
using std::endl;

void printone()
{
    cout << "\n8 3 4"<<"\n1 5 9"<<"\n6 7 2";
};
void printtwo()
{
    cout << "\n6 1 8"<<"\n7 5 3"<<"\n2 9 4";
};
void printthree()
{
    cout << "\n4 9 2"<<"\n3 5 7"<<"\n8 1 6";
};
void printfour()
{
    cout << "\n2 7 6"<<"\n9 5 1"<<"\n4 3 8";
};

int _tmain(int argc, _TCHAR* argv[])
{
    char loc;
    int num;
    cout << "\nenter location code and number: ";
    cin >> loc >> num;
    if (num < 1 || num > 9)
    {
        cout << "\nerror in input";
        return 0;
    }
    if (loc == 'e')
    {
        if (num == 5)
        {
            printone();
            return 0;
        }
        else
        {
            cout << "\nmagic square not possible";
            return 0;
        }
    }
    if (loc == 'a' || loc=='c' || loc=='g' || loc=='i')
    {
        if (num%2 == 1) // mod operation to check for odd number
        {
            cout << "\nmagic square is not possible";
            return 0;
        }
        if ((loc=='a'&&num == 8)||(loc=='c'&&num == 4)
            ||(loc=='g'&&num == 6)||(loc=='i'&&num == 2))
        {
            printone();return 0;
        }
        if ((loc=='c'&&num == 8)||(loc=='i'&&num == 4)
            ||(loc=='a'&&num == 6)||(loc=='g'&&num == 2))
        {
            printtwo();return 0;
        }
        if ((loc=='g'&&num == 8)||(loc=='a'&&num == 4)
            ||(loc=='i'&&num == 6)||(loc=='c'&&num == 2))
        {
            printfour();return 0;
        }
    }
```cpp
if (loc == 'a' || num == 2)
    printfour(); return 0;
}
if (loc == 'b' || loc == 'd' || loc == 'f' || loc == 'h')
{
    if (num % 2 == 0) // mod operation to check for odd number
    {
        cout << "\nmagic square is not possible";
        return 0;
    }
    if ((loc == 'b' && num == 3) || (loc == 'd' && num == 1)
        || (loc == 'f' && num == 9) || (loc == 'h' && num == 7))
    {
        printone(); return 0;
    }
    if ((loc == 'f' && num == 3) || (loc == 'b' && num == 1)
        || (loc == 'h' && num == 9) || (loc == 'd' && num == 7))
    {
        printtwo(); return 0;
    }
    if ((loc == 'd' && num == 3) || (loc == 'h' && num == 1)
        || (loc == 'b' && num == 9) || (loc == 'f' && num == 7))
    {
        printthree(); return 0;
    }
    if ((loc == 'h' && num == 3) || (loc == 'f' && num == 1)
        || (loc == 'd' && num == 9) || (loc == 'b' && num == 7))
    {
        printfour(); return 0;
    }
    cout << "\nerror - no match";
    return 0;
}```