British Informatics Olympiad

The 2004 British Informatics Olympiad

Time allowed: 3 hours

LIONHEAD STUDIOS

Instructions

You should write a program for part (a) of each question, and produce written answers to the remaining parts. Programs may be used to help produce the answers to these written questions but are not always necessary.

You may use a calculator and the on-line help that your programming language provides. You should have a pen, some blank paper, and a blank floppy disk on which to save your programs. You must not use any other material such as disks, files on a computer network, books or other written information. You may not communicate with anyone, other than the person invigilating this paper.

Mark the first page of your written answers with **your name, age in years** and **school/college**. Number all pages in order if you use more than one sheet. All of your computer programs should display your name and school/college when they are run, and the floppy disk you use to submit the programs should also show your name and school/college.

For your programs to be marked, the source code must be saved, along with executables if your language includes a compiler; this includes programs used to help answer written questions. You must clearly indicate the name given to each program on your answer sheet(s).

Sample runs are given for parts 1(a), 2(a) and 3(a). **`Bold text`** indicates output from the program, and `normal text` shows data that has been entered. The output format of your programs should follow the 'sample run' examples. Your programs should take less than **10 seconds** of processing time for each test.

Attempt as many questions as you can. Do not worry if you are unable to finish this paper in the time available. Marks allocated to each part of a question are shown in square brackets next to the questions. Partial solutions (such as programs that only get some of the test cases correct within the time limit, or partly completed written answers) may get partial marks. Questions can be answered in any order, and you may answer the written questions without attempting the programming parts.

Hints

- If you can only see how to solve part of a problem it is worth writing a program that solves that part. We want to give you marks and questions are evaluated using multiple tests of differing difficulty. Remember, *partial solutions may get partial marks*.
- Question 2 is an implementation challenge and question 3 is a problem solving challenge.
- Most written questions can be solved by hand without solving the programming parts.
- Do not forget to indicate the name given to your programs on your answer sheet(s).

Question 1
Mayan Calendar

The Mayan civilisation used three different calendars. In their *long count calendar* there were 20 days (called **kin**s) in a **uinal**, 18 uinals in a **tun**, 20 tuns in a **katun** and 20 katuns in a **baktun**. In our calendar, we specify a date by giving the day, then month, and finally the year. The Maya specified dates in reverse, giving the baktun (1-20), then katun (1-20), then tun (1-20), then uinal (1-18) and finally the kin (1-20).

The Mayan date **13 20 7 16 3** corresponds to the date **1 January 2000** (which was a Saturday).

1 (a)
[24 marks]

Write a program which, given a Mayan date (between **13 20 7 16 3** and **14 1 15 12 3** inclusive), outputs the corresponding date in our calendar. You should output the month as a number.

Sample run

```
13 20 9 2 9
22 3 2001
```

You are reminded that, in our calendar, the number of days in each month is:

1	January	31
2	February	28 / 29 (leap year)
3	March	31
4	April	30
5	May	31
6	June	30
7	July	31
8	August	31
9	September	30
10	October	31
11	November	30
12	December	31

Within the range of dates for this question, every year divisible by 4 is a leap year.

1 (b)
[2 marks]

What are the Mayan dates for **1 February 2000** and **1 January 2001**?

1 (c)
[3 marks]

The Maya believed the universe was destroyed and re-created every *cycle* of their calendar, which was 20 baktun in length. How many kins (days) are there in a cycle? What day of the week is the last day of the current cycle (**20 20 20 18 20**)?

Question 2
Four in a Line

In the game of *Four in a Line*, two players alternate dropping pieces of their own colour into a vertical board. The board consists of seven columns, each of which can hold six pieces. On their turn, a player can choose to drop their piece into any column which still has an empty space. When a piece is dropped into a column, it falls to the lowest unused position in that column.

The winner is the first player to form a line of four adjacent pieces of their own colour; horizontally, vertically or diagonally.

A simple strategy is for a player to choose their moves based on the following rules:

1. If there is a move which will win the game immediately, this move is played. If several such moves exist, the leftmost one is played.
2. If rule 1 does not indicate which move to take and there is a move which, if played by the other player would cause them to win immediately, this move is played. If several such moves exist, the leftmost one is played.
3. If neither rule 1 nor rule 2 indicates which move to take, play in the leftmost column which still has empty space.

The following examples demonstrate the simple strategy on some typical boards. Note that the boards themselves have arisen from an initial sequence of moves that did not follow the strategy.

The first player's pieces are shown using ① and the second player's pieces are shown using ❷. The columns are numbered from left to right, with the leftmost column being number 1.

If it is the first player's turn on the following board, they must play in column 3, according to rule 1, to win with a horizontal line of four pieces.

			❷	❷	❷	
			①	①	①	

If it is the second player's turn on the following board, they must play in column 4, according to rule 2, to prevent the first player winning with a diagonal line on their next move by playing in this column.

①						
❷	①					
❷	❷	①			①	
❷	❷	①			①	

If it is the first player's turn on the following board, they must play in column 1, according to rule 3.

	①	❷	❷			
	❷	①	①			

2 (a)
[26 marks]

Write a program to play Four in a Line. Both players will be controlled by the computer and play using the simple strategy.

Your program should first read in a single integer, n ($1 \le n \le 10$). This will be followed by a line containing n integers, indicating the columns for the first few moves of the game; the first number indicating the first move for player one, the next number indicating the first move for player two, etc. The columns are numbered from left to right, with the leftmost column being number 1. None of these initial moves will generate a winning line.

You should then output the board, indicating empty spaces with a `-`, pieces belonging to player 1 with a `*` and pieces belonging to player 2 with an `o`.

Until your program terminates you should repeatedly wait for input, and then:

- If you receive the letter `n`, you should play the next move of the game.
- If you receive the letter `r`, you should play the rest of the game, until one player has won or the board is full.
- Ignore any other input

After each command, you should output the board. If the game has been won you should output **`Player 1 wins`** (for a first player win) or **`Player 2 wins`** (for a second player win), and then terminate. If the board is full, you should output **`Draw`** and then terminate.

Sample run

```
9
3 2 3 4 4 4 2 2 3

-------
-------
-------
-o*o---
-***---
-o*o---

n
                    [Rule 2]
-------
-------
--o----
-o*o---
-***---
-o*o---

n
                    [Rule 3]
-------
-------
--o----
-o*o---
-***---
*o*o---

r
          [Rule 2 then Rule 1]
-------
-------
--o*---
-o*o---
o***---
*o*o---

Player 1 wins
```

2 (b)
[2 marks]

What is the final board position, playing until one player has won or the board is full, if both players follow the strategy, starting with an empty board?

2 (c)
[6 marks]

The following board has arisen after a game in which the two players, using an unspecified strategy, continued to play after the game had been won:

```
o**oo**
*oo**oo
o**oo**
*o***oo
o*o*o**
*oo*ooo
```

What is the smallest number of moves which might have occurred when a winning position was reached? What is the largest number of moves? Justify your answer for the smallest number of moves.

2 (d)
[3 marks]

Suppose player one drops all 21 of their pieces into the board, without player two playing any of their pieces. How many different board patterns can be generated?

Question 3
Morse Code

The BIO has been receiving telegrams congratulating it on reaching its 10th anniversary. At least, we think it has. The telegrams have been sent in *Morse code* and, unfortunately, the gaps between letters have been left out.

In Morse code, each letter of the alphabet is replaced by a sequence of dots and dashes as follows:

a	.-	h	....	o	---	v	...-
b	-...	i	..	p	.--.	w	.--
c	-.-.	j	.---	q	--.-	x	-..-
d	-..	k	-.-	r	.-.	y	-.--
e	.	l	.-..	s	...	z	--..
f	..-.	m	--	t	-		
g	--.	n	-.	u	..-		

Every combination of between 1 and 4 dots and dashes is used, except for:

```
..--
.-.-
---.
----
```

Traditionally, dots were transmitted by a short note and dashes by a longer note, with pauses between different letters. This is why some mobile phones make the sound `... -- ...` when receiving a message, since this is the Morse code for SMS.

If the gaps between letters are missed out, messages can be ambiguous. For example, even if we know the message `-..------.` is made up of three letters, it might mean **njg**, **dog**, **xmg** or **xon**.

3 (a)
[26 marks]

Write a program which reads in a message (between 1 and 10 letters inclusive) and determines how many messages, *with the same number of letters as the input*, it might represent.

Sample run

```
dog
4
```

3 (b)
[5 marks]

How many messages might `-----` represent, if we do not know the number of letters in the message? How about `-..------.`?

3 (c)
[3 marks]

It is possible to come up with new ways of encoding the alphabet so that, even when the gaps between letters are missing, messages are unambiguous. The *size* of such an unambiguous encoding is the total number of dots and dashes in a message containing each letter once.

For example, we could encode each letter by some dots (indicating its position in the alphabet) followed by a dash; so `.-` would be **a**, `..-` would be **b**, and 26 dots followed by a dash would be **z**. This encoding has a size of 377 (2 + 3 + ... + 27).

What is the smallest size an unambiguous encoding can have?

Total marks: 100

End of BIO 2004 Round One paper