British Informatics Olympiad

The 2002 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.

Mark the first page used for 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 **30 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. 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
Lojban

Counting in **Lojban**, an artificial language developed over the last forty years, is easier than in most languages. The numbers from zero to nine are:

1	pa	4	vo	7	ze		
2	re	5	mu	8	bi	0	no
3	ci	6	xa	9	so		

Larger numbers are created by gluing the digits together. For example, 123 is **pareci**.

1 (a)
[22 marks]

Write a program that reads in a Lojban string (representing a number less than or equal to 1,000,000) and outputs it in numbers.

Sample run

```
Lojban: renonore
Number: 2002
```

1 (b)
[2 marks]

What is **sovo** + **rexa**? Give your answer in Lojban.

1 (c)
[4 marks]

A **self-descriptive** number is equal to the sum of the positions of its letters in the alphabet. For example, **pareci** is not self-descriptive, since 123 is not equal to 16 + 1 + 18 + 5 + 3 + 9 (=52). Give a self-descriptive Lojban number.

Question 2
Shuffling

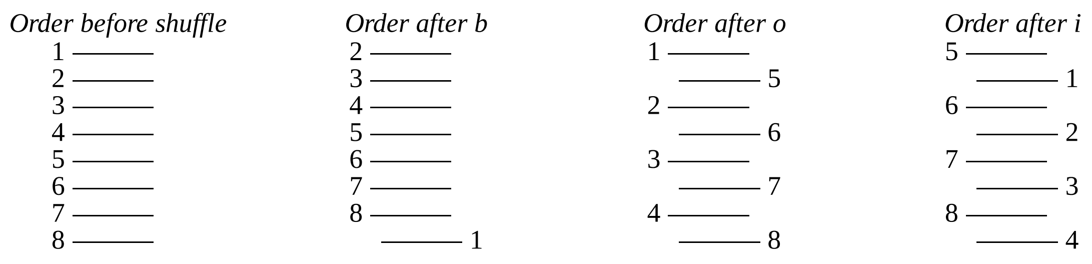
Before playing a card game you usually shuffle the cards to mix them up, but how jumbled do they really become after a few shuffles? Your task for this question is simulate the shuffling of a pack of cards.

The simulation will be on a pack of eight cards, starting with the cards in the order (from top to bottom) 1, 2, 3, 4, 5, 6, 7 then 8.

There are three basic shuffles to be modelled:

The first shuffle is the **break** (**b**), where the top card is simply placed on the bottom of the pack. After this shuffle the cards would be in the order 2, 3, 4, 5, 6, 7, 8 then 1.

The next two shuffles are **riffles**. In these shuffles the pack is split into two equal halves (the top four cards and the bottom four cards) and then the two halves are interleaved. In the **out riffle** (**o**) the card that was previously top stays top; after this shuffle the cards would be in the order 1, 5, 2, 6, 3, 7, 4 then 8. In the **in riffle** (**i**) the card that was previously top will become the second card; after this shuffle the cards would be in the order 5, 1, 6, 2, 7, 3, 8 then 4.



The three basic shuffles can be combined to form more complex shuffles. Shuffles are described as follows:
1. The three basic shuffles are denoted by the letters **b**, **i** and **o**.
2. The shuffles are written in the order they should be done.
3. A number can be put before a basic shuffle to indicate it should be done several times.
4. A complex shuffle can be put in brackets and a number put before the brackets, indicating the complicated shuffle should be done several times.

For example:
- **io** is the in riffle followed by the out riffle.
- **4b** is a sequence of four breaks.
- **4bio** is a sequence of four breaks, followed by an in riffle then an out riffle.
- **4(bio)** means repeat the sequence "break, in riffle, out riffle" four times.

2 (a)
[25 marks]

Write a program that shuffles a pack of eight cards.

Your program should first read in a single line, containing a description of the shuffle. This line will have less than 20 characters, and any numbers will be between 2 and 9 inclusive.

Descriptions will meet rules 1-4. In other words, all numbers will be followed by a letter or bracket, and all brackets will contain complex shuffles.

You should apply the shuffle to a pack of eight cards (starting in the order 1, 2, 3, 4, 5, 6, 7 then 8) and then output the final order of the cards.

Sample run 1

```
i2o
5 6 7 8 1 2 3 4
```

Sample run 2

```
3(b2(oi))
1 6 3 2 4 5 7 8
```

2 (b)
[2 marks]

What would be the result of applying the shuffle **bio** to a pack of 20 cards that are numbered 1–20 and initially are in the order 1, 2, …, 20?

2 (c)
[3 marks]

After how many in riffles will a pack of 8 cards first return to the order before shuffling started? After how many out riffles? After how many breaks?

2 (d)
[7 marks]

Is it always possible to return a pack of cards, whatever its size, to its original order (i.e. the order before shuffling started) by just using the in riffle? Justify your answer.

Note that the in and out riffles are also possible with an odd number of cards. For example, the pack 1, 2, 3, 4, 5 would become 1, 4, 2, 5, 3 after an out riffle, or 3, 1, 4, 2, 5 after an in riffle.

Question 3
Mops

A **mop expression** is way of producing a number using only multiplications, ones and pluses. The **length** of a mop is the number of ones it contains.

For example, 22 = 1+1+ ((1+1+1+1) $\times$ (1+1+1+1+1)), which has a length of 11.
Another mop of 22 is 1+ ((1+1+1) $\times$ (1+ ((1+1)$\times$(1+1+1)))) which has a length of 10.
There are no shorter mops for 22.

3 (a)
[24 marks]

Write a program that inputs a single integer *n* ($1 \le n \le 10000$) and outputs the length of the shortest mop of *n*.

Sample run

```
22
10
```

3 (b)
[3 marks]

Give a mop equal to 100, with length 16.

3 (c)
[3 marks]

If you have found a mop equal to *n*, with length *l*, is it always possible to find a longer mop that is also equal to *n*? Justify your answer.

3 (d)
[5 marks]

What is the highest value a mop of length 44 can have?

Total marks: 100.

End of BIO 2002 Round One paper