Problems for OpComp 2024

By Dr. Robert W. Hasker
Friday 22 November 2024

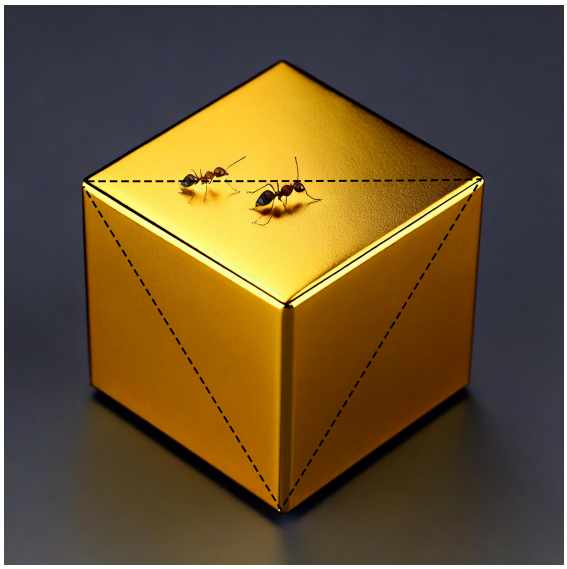
Notes:

- Unless a problem specifies otherwise, you may assume that all user inputs are valid.
- The precise wording of your output is not critical as long as it is completely clear and meets any stated requirements.
- There is generally no runtime limit for solutions, but if a solution runs far longer than deemed reasonable it will be terminated and marked as incorrect.

See the OpComp website for additional instructions and rules.

1. Exploring Ants (10 points)

Ants have found a cave of crystals. Each crystal is a rectangular solid; that is, each has 6 sides intersecting at perfect right angles, and each is buried in the ground with one of the corners pointing up. This means that just three faces (and four corners) are showing, so the ant can explore each face in a path like the one shown with a dashed line below:



Your program will read the length of each of the sides and compute how far an ant would travel on the dashed line path visiting all three faces. Read the height, width, and length (with no commas or other non-space characters between) and print the total distance covered. Assume all distances (both input and output) are in centimeters.

2. Cars Afield (10 points)

An ASCII artist uses standard keyboard characters to draw pictures on the screen. A particular artist draws ASCII pictures of fields with cars parked in them. The only visible part of the car is the headlights, shown as two o's one space apart. Thus "oio" and "o.o" show car placement, but "ooo" must be something else since modern cars do not have headlights in the middle. Capital O's are extra bright headlights - "hi beams" - so cars can have two high beam lights instead of two low beam lights. As in the real world, headlights belong to just one car.

Your program will read a single line of input (which can contain spaces) and output the number of cars found. For example, "`..vvXvo,o…||0i0`" contains two cars.

3. Prime Birth Years (10 points)

King Primo adores prime numbers so gives a special new year's gift to every family in years where a parent has circled the sun a prime number of times and the same can be said for the child. To help the family pick the parent and child, you write a program to ask about the child/parent gap and print a number of gifts and the years of the last gift. Due to excellent health care in this kingdom, assume everyone lives to 100. You must follow this example output:

```
A 52 year difference gives 4 gifts with the last at 37 and 89.
```

Zero gifts are the same but without ages. The parent must be older than the child.

4. Happy Alphabetical Ants (20 points)

Grover's ants like to walk across sentences, finding letters next to each other which follow alphabetical ordering. They are happiest at the longest section. Write a program to read a line of text and write out the segment that results in the happiest ants. Ignore punctuation and spaces as shown by the following (with user input in bold face) and print the text's position based on the original input. Number the first character as #1.

```
The quick, red fox jumped over the lazy, brown dog
Happiest: dfox starting at character 14
```

Note that skipping letters in the alphabet (eg, going from d to f) does not make the ants sad. Break ties by the first time the ants are at their happiest.

5. How Long Until We Meet? (20 points)

Two ant colonies live in a cave, Romeant on the ceiling and Juliant on the floor. They want to meet, but have to wait until the stalactites on the ceiling grow to meet the stalagmite on the floor. Through careful study, they have found that both the stalactites and stalagmites grow relatively fast at 0.2 mm per year. Growth rates are constant over long time periods, so the colonies wish to find out how long they must wait before one of the stalactites meets one of the stalagmites.

Juliant has mapped the floor, measuring the height of each point in centimeters from the lowest point. They used ASCII letters a-z to represent the the height of each square in millimeters with a=0, b=1, c=2, and so on to z=25. Romeant has done the same, but that colony has measured downwards from the highest point. From other data the colonies know the distance from the highest point on the ceiling to the lowest point on the floor (that is, the vertical distance between surfaces marked as "a" in both maps), so they can use this to find how close the stalactites are to the stalagmites. Your program is to print the time in years for some part of the ceiling to meet some part of the floor; that is, where a stalactite will first meet a stalagmite.

The input to your program is in three parts. First, the distance between the highest ceiling point and the lowest floor point in centimeters. Second, the number of rows in each map followed by the number of columns. This is followed by the map data collected by Romeant (listed in rows, followed by a blank line, followed by the map data collected by Juliant. Both of these are organized so that each point in Romeant's map is directly above the corresponding point in Juliant's map. You are guaranteed each point is a letter between a and z and that the input is formatted correctly. In particular, the number of rows and columns in each map is correct. The program will prompt the user for the data file name and read the information from that file. It will print the number of years before meeting, rounded up to the next whole number of years.

For example. if a file contains

```
20
3
6
abcdef
acaeaa
babbgg

aabbaa
acdcdi
begjgd
```

then the output would be

```
Number of years: 400
```

6. Nested Boxes (20 points)

The holiday elves are hard at work making gifts and have little time for cleanup. But they plan to reduce their carbon footprint by reusing empty cardboard boxes. They have purchased a robot to nest empty boxes into stacks and they have set up a conveyor belt to send the boxes to the robot. The robot needs to be programmed to take the first box and stack as many other boxes inside the first as possible. When a box does not fit into the current stack, the robot will set the stack aside and start creating another. It will process boxes in the order they arrive; it will never skip a box.

A box fits inside another if it is shorter (has less height) and if it can be rotated so both the width and length are smaller. The boxes will always be nested inside each other; the robot will never fit two boxes side-by-side in a larger box. Box dimensions are specified in centimeters (using floating

point), and for a box to fit in a particular direction (height, width, length), it must be at least 1 centimeter smaller that direction.

Write a program which reads a series of boxes from input. Each box is on its own line, consisting of a name followed by the dimensions. The dimensions are separated by 'x', For example, a helmet box 12 centimeters wide, 13 centimeters deep, and 9.5 centimeters high would be listed as

```
helmet 12x13x9.5
```

The program is to read box information from the user until the three dimensions are 0. It will print each stack as it is created. Stacks are printed on a single line with the boxes in that stack listed in order from largest to smallest, separated by commas. For each box, print the name and dimensions as they were read from input. For example, if the input is

```
bigbox 55x45x45
boot 43x40x40
shoe 32x18x19
soda 17x17x17
ball_1 20x20x12.0
ball_2 18.0x18x18.5
glove 5x5x17.4
ignore 0x0x0
```

then the output would be

```
bigbox: 55x45x45, boot: 43x40x40, shoe: 32x18x19, soda: 17x17x17
ball_1: 20x20x12.0
ball_2: 18.0x18x18.5, glove: 5x5x17.4
```

Note `ball_1` and `ball-2` include the `.0` after certain dimensions.

7. Boxed Gifts (40 points)

Santa is in competition with many other holiday mascots, so he has asked his elfin helpers to improve the presentation of under the tree. One of the elves took a discrete mathematics course and learned that rectangles are judged more pleasing when the ratio of the sides follows the "golden ratio". This ratio is (1 + sqrt(5))/2, or about 1.618. The golden ratio appears in many ways in paintings and architecture. Through careful research, the elves have found that families are happiest if the presents are arranged under the tree so that the boxes in front of the pile follow the golden ratio. You have been asked to write a program that reads package information and finds the largest collection of boxes meeting this criterion.

In particular, your program is to read box data as described in problem 6 (Nested Boxes) and then print the largest collection of boxes which follow the golden ratio in their size differences. Whereas box sizes were treated as integers in problem 6, they are treated as real numbers in this problem. Compute the size of a box as its total volume. The sizes of two boxes to meet the golden ratio criterion if one size is between 1.55 and 1.65 of the other's size. Ignore any items which have zero volume; these are sheets of paper. Unlike problem 6, the input to 7 ends with the end of input, not

with a box with dimension 0x0x0. If two collections of boxes meeting the golden ratio contain the same number of boxes, print the collection with the larger primary box. When printing the chosen list, print the boxes in order by total size from largest to smallest. For each box, print the name and the dimensions. If there are no pairs of boxes meeting the golden ratio, print a message to that effect.

8. Embedded Words (40 points)

You've been hired to be the puzzlemaster on NPR's Weekend Edition Sunday, and you are trying to build a puzzle based on finding words hidden in other words. You have decided to write a program that takes a word, looks it up in a dictionary, and prints all words that match a subset of its letters in sequence. That is, for a word X to be embedded in another (word Y), the in X must match, in sequence, the letters of Y after ignoring at least one letter in X. As an example, "and" matches both "plant" (ignoring the 'd') and "address" (ignoring the 'n'). The plan is to then use this information to develop an on-air puzzle.

Some further details:

- Suppose you have an target word (X) and a dictionary word (Y). If X is embedded in Y, then the embedding is "trivial". Do not include trivial embeddings in the output.
- Cases in which the dictionary word Y is embedded in the target are also considered trivial (and so skipped).
- Print just the longest embeddings; if there is an embedding of length N (that is, N letters from the target word are found in a dictionary words), then embeddings for lengths less than N are ignored. However, you will print all embeddings of length N.
- Do not consider embeddings for single characters or empty strings. If there are no larger embeddings, print "`No embeddings found`".
- The list of embedded words is to be sorted alphabetically.

Your input will be a word followed by the dictionary to use for that run. Each word is on its own line. For example, if the input is

```
grand
are
bandersnatch
bangle
branch
breakfast
grandfather
great
hand
plant
```

then the output will be

```
bandersnatch
branch
great
hand
```

but if the first word is changed to “zoo”, the output would be

```
No embeddings found
```

You can assume there is a target word in every input and that every dictionary contains at least one word.

9. Finding Ant Food (40 points)

The ant colony Juliant finds a new cave and learns there is food in the cave. Unfortunately there are also spiders, and these spiders eat ants. Through various expeditions, Juliant has mapped the cave into 10-centimeter square grids with each grid position marked with ‘–’ if it is empty, ‘f’ if it contains food, ‘s’ if it contains a spider, and ‘>’ if it is the entrance. Write a program which prompts the user for a file, reads the map from that file, and prints the minimum number of spiders that must be passed to reach the food. This will allow Juliant to calculate how many ants to send to the food to ensure some food returns in spite of the losses. Each line in the input file is the same length and contains only legal map characters. You can assume the grid is no more than 30 cells in any direction, there is always an entrance, and there is always food. Assuming an ant is in a cell that is not next to a wall (that is, the edge of the map), it can move in any of the 8 directions that surround the cell including diagonally.

For example, the test data file `9a.txt` (available on the contest website) contains the following map:

```
--ss-sf--
-ss--s---
>sss---ss
--s-sss--
```

This has two paths which cross just one spider. One path goes past the first spider in the top row, the other goes past the second spider (from the left) in the second row.