

High School Programming Competition

Sponsored by Fast Enterprises, Clearwater Advisors, BaliHoo Inc and
MyInteractions.com

19th April 2008

Organized by the Department of Computer Science,
College of Engineering,
Boise State University

Bart's Skateboard Park

Description: Bart is skateboarding through the town of Springfield and wants to find the 3 block section of town with the most jumps to designate as his own skateboard park.

Input is number of blocks followed by a list of block descriptions. On each line is the block number, followed by the number of jumps on that block. The block numbers start at 1. If descriptions are provided for n blocks, you may assume they are for blocks 1 through n . You may assume $n \geq 3$. Note the block numbers are not necessarily sorted.

Output is the 3 consecutive blocks with the most jumps (with blocks listed in sorted order according to block number). In case of ties, we want the lowest numbered three blocks.

Examples: Example 1

Input:

```
5
1 2
2 6
3 3
4 4
5 1
```

Output: 2 3 4

Example 2

Input:

```
5
4 4
2 6
5 1
1 2
3 3
```

Output: 2 3 4

Problem 2: Character Frequencies

Description: We want to be able to count the frequency of occurrence of letters of the English alphabet in a given file given as a command line argument. Your job is to read the file and count the occurrence of the letters a through z (we will ignore the case). After reading the file, we want to sort the letters by their frequencies and then output the frequencies in descending order. Three sample files have been provided in your folder: Bill-of-Rights.txt, Alice-in-Wonderland.txt, Complete-Shakespeare.txt.

Examples: `java CharacterFrequencies Bill-of-Rights.txt`

```
e = 1047
t = 836
o = 670
i = 602
r = 567
a = 544
n = 516
s = 490
l = 325
h = 304
c = 280
d = 278
u = 255
y = 211
p = 210
m = 196
f = 187
b = 160
g = 143
w = 87
v = 73
x = 66
j = 33
k = 24
q = 13
z = 5
```

Problem 3: Spell Checking

Description: Write a program that will check the correctness of a given list of words using a known dictionary of all correct words in all their forms.

If the word is absent in the dictionary then it can be replaced by correct words (from the dictionary) that can be obtained by one of the following operations:

- deleting of one letter from the word;
- replacing of one letter in the word with an arbitrary letter;
- inserting of one arbitrary letter into the word.

Your task is to write the program that will find all possible replacements from the dictionary for a list of given words. The words to be checked are read from the standard input.

The dictionary is provided as a file. The name of the dictionary should be passed in as a command line argument to your program. Two sample dictionaries are provided in your home folder. These are named `test-dictionary` and `large-dictionary`.

All words in the input file (words from the dictionary and the words to be checked) consist only of lowercase alphabetic characters and each one contains 50 characters at most.

Examples: Sample dictionary:

```
i
is
has
have
be
my
more
contest
me
too
if
award
```

Inputs:	Outputs:
me	me is correct
aware	aware: award
m	m: i my me
contest	contest is correct
hav	hav: has have
oo	oo: too
or	or:
i	i is correct
fi	fi: i
mre	mre: more me