

# ***Software Development Homeworks Discussion***

*Davide Balzarotti*

Eurecom – Sophia Antipolis, France

# It's Over

- 64 Registered Users
- 11 Weeks
- 2350 Submissions
- 522 Correct solutions
  - 15% since sunday afternoon!!
- The graphs in the rest of the talk are updated to sunday afternoon

**And the Winner is.....**



**bancel**

```
root@students:/var/hwe# game rank
```

```
-----  
Global Ranking:  
-----
```

```
3065 - bancel  
3035 - pxthanh  
3031 - haradwaith  
3018 - eeko  
3013 - rogdham  
3007 - lucab  
3004 - daffes  
3003 - rioult
```



# Firsts to Solve the Challenges

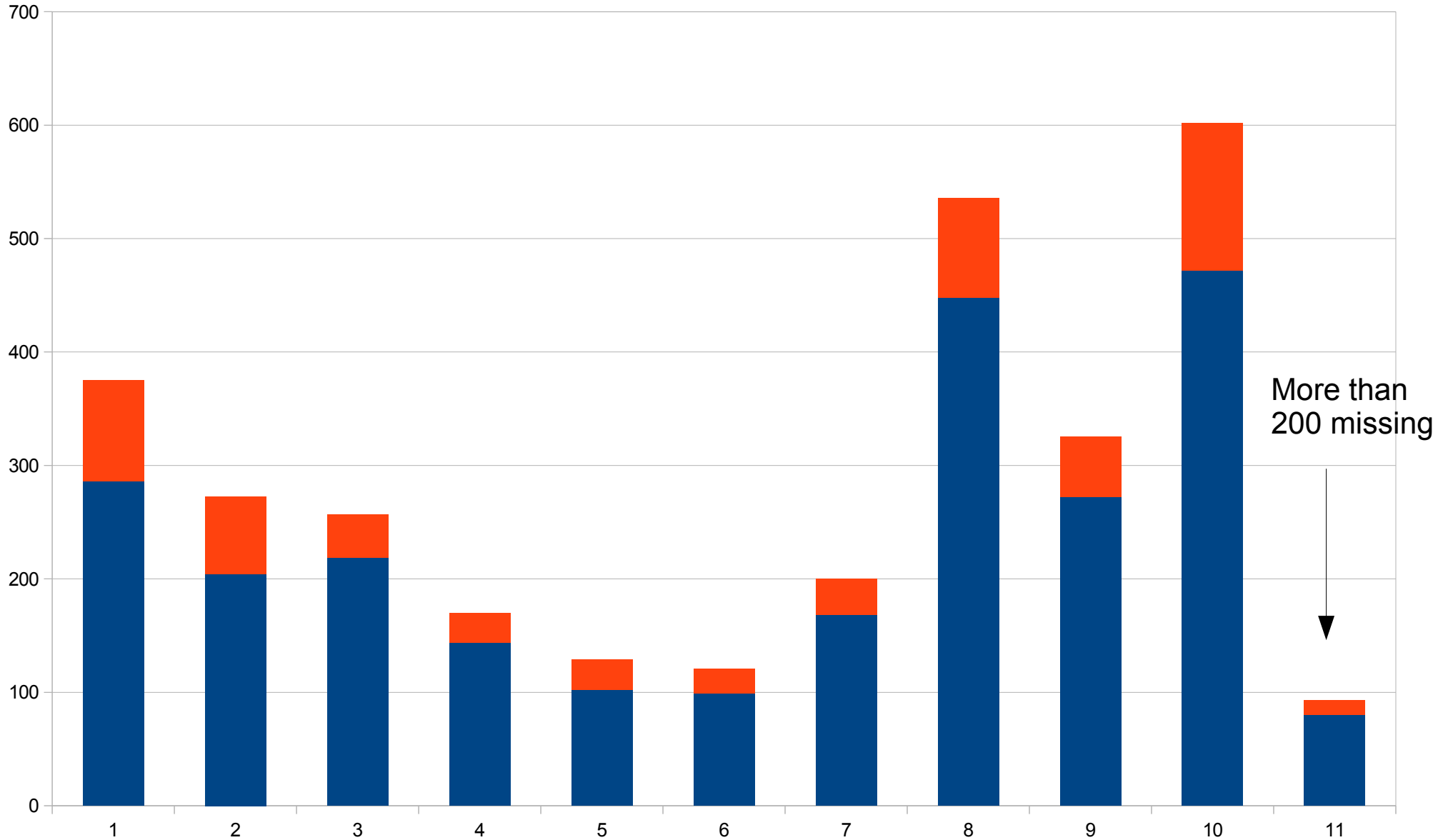
bancel			
pxthanh			
haradwaith			
eeko			
rogdham			
lucab			
rioult key20 daffesl			



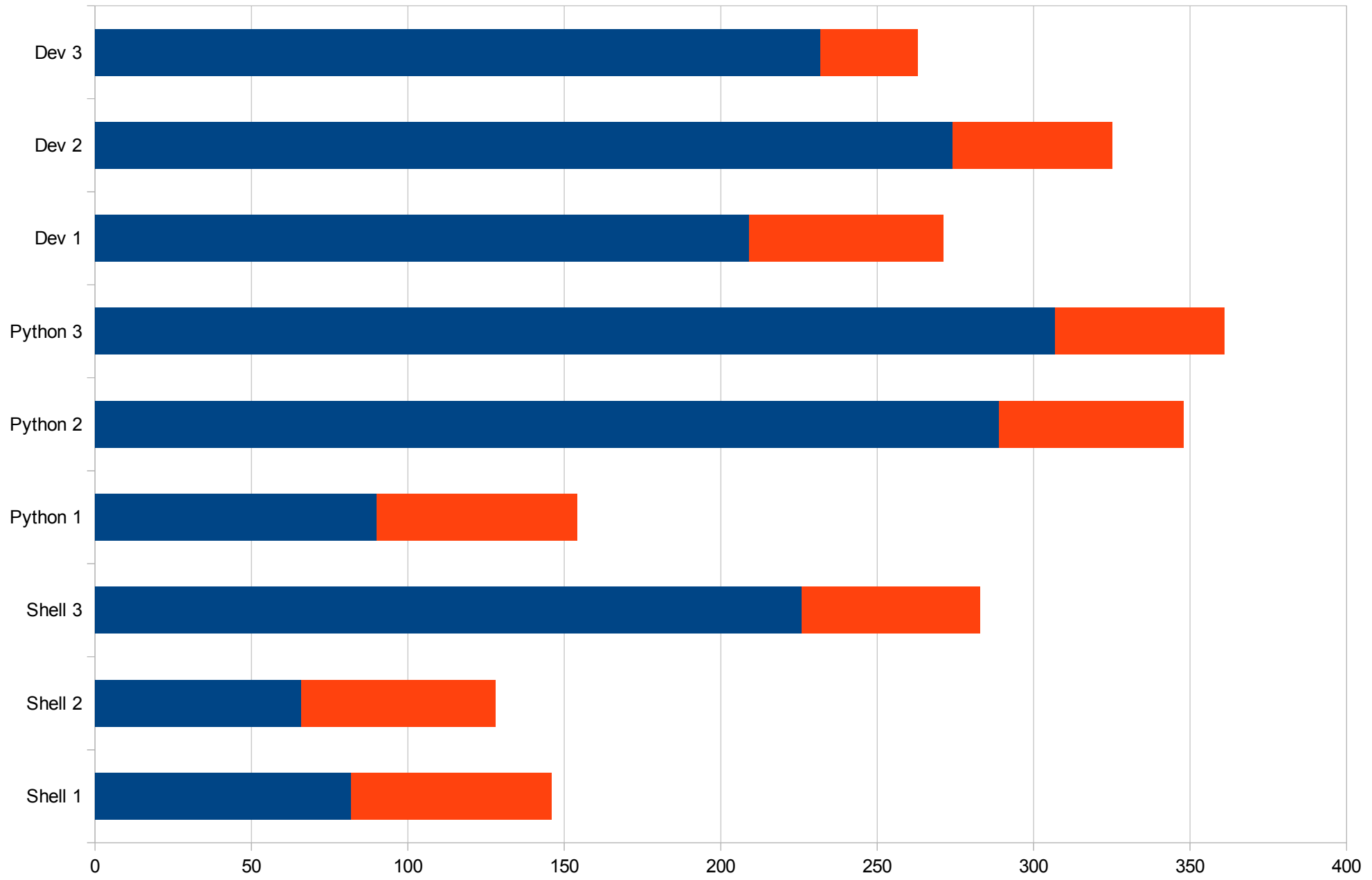
# Firsts to Solve the Challenges

bancel			
pxthanh			
haradwaith			
eeko			
rogdham			
lucab			
rioult key20 daffesl			

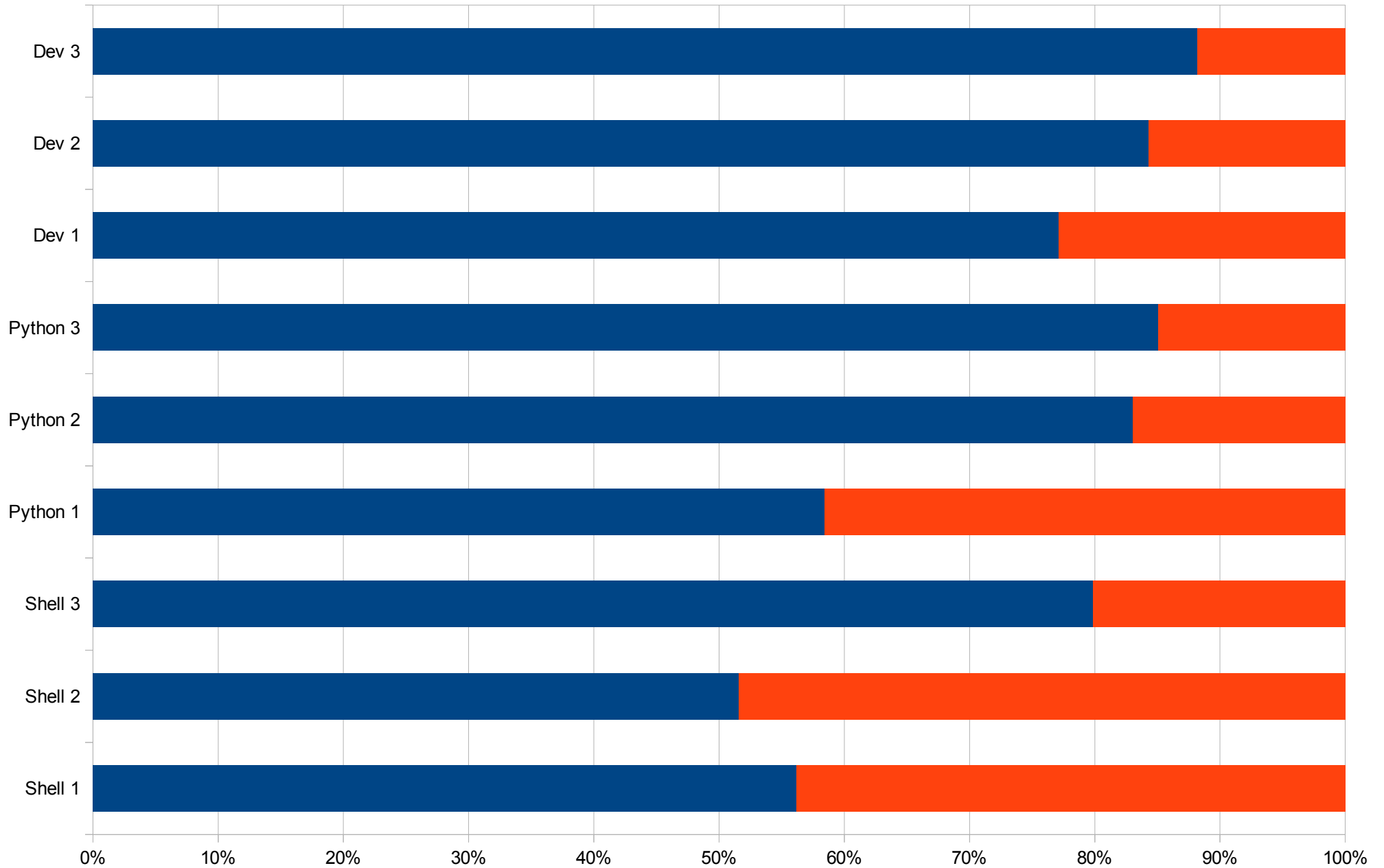
# Stats: Submissions per Week



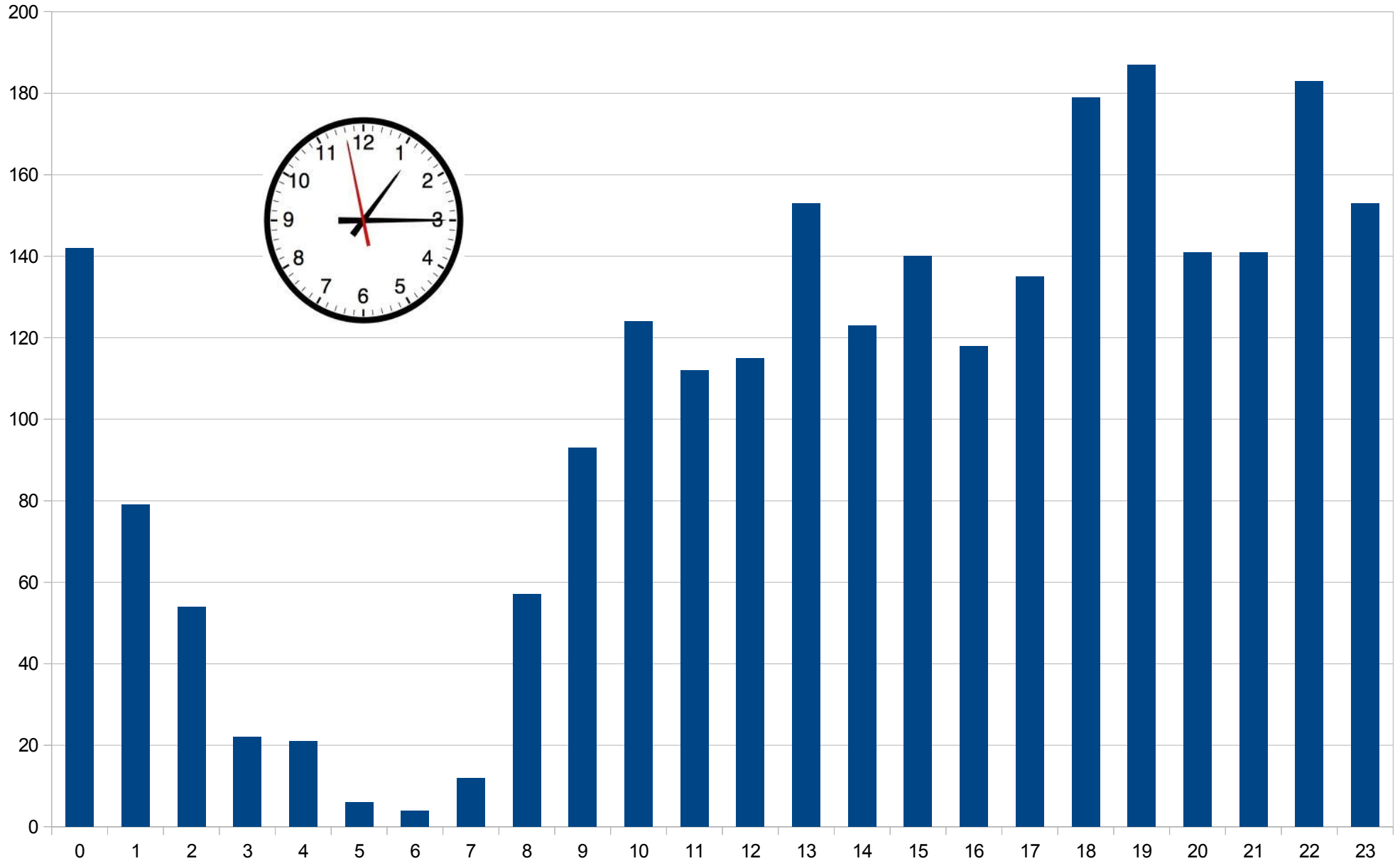
# Stats: Submissions per Challenge



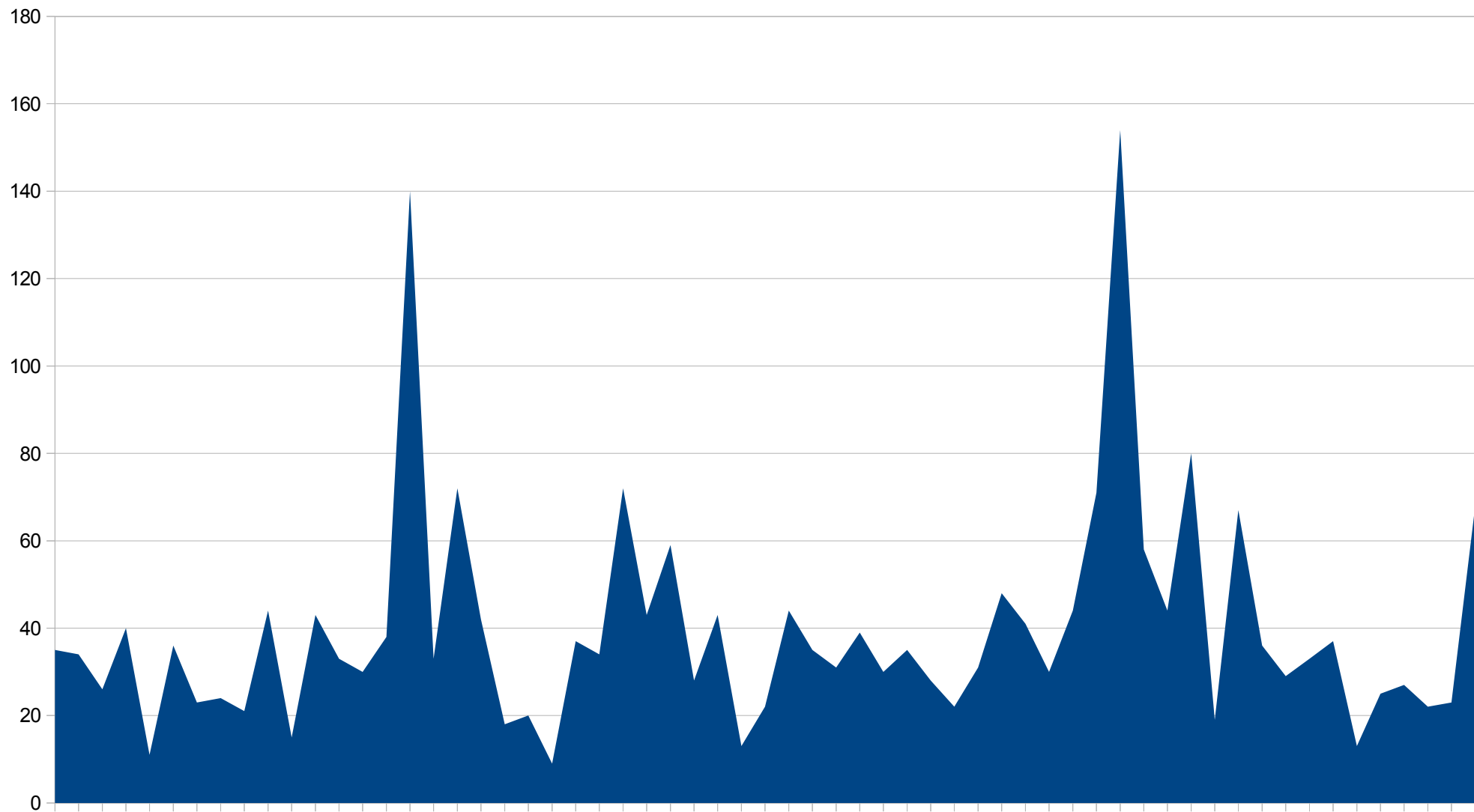
# Resubmission Ratio



# Submission by Time of the Day



# Stats: Average Submissions per User



# Command-Line 1

One day you download a blacklist of domains that are known to provide malicious content. (you find an example of it in the file `blacklist.txt` in the current directory).

At that point you start wondering if, in the past, you visited some of these entries...

You want to print the domains in "`blacklist.txt`" that appear in the "`urls.txt`" file (a snapshot of the urls visited by your browser in the last 6 months).

- Stats:
  - Solved by: ALL
  - Total submissions: 146
  - First to solve: pxthanh

```
ww2.bizitangel.ru  
ynnyouemrnsqhrf.info  
www.webro.ro
```

```
http://www.redchili.byethost12.com/index4.php  
http://www.3all.co.il/web/Sites/clubpenguin/PAGE107.asp  
ftp://www.webro.ro/root/cc  
http://www.facebook.com/index.html  
http://web.tiscali.it/xnhatto/  
http://www.redchili.byethost12.com/index5.php  
http://www.all5dr.org/fichiers/44/plaquette/p12.htm  
http://damnurl/x.php?par=ynnyouemrnsqhrf.info&check=True  
http://fawebro.ro/tricky/eheh.jpg  
http://ww2.bizitangel.ru/
```

```
> sort blacklist.txt  
<(cut -d/ -f3 urls.txt | sort | uniq) |  
uniq -d
```



Somebody did it in 69 lines of code and 2055 characters !! :(

# Development 1

In this exercise, you have to automate the building process of a strange application that contains a different C file every month. For example, to build the app in January you would run:

```
> gcc -c January.c -o January.o  
> gcc strange_app.c January.o -o strange_app
```

If instead you build it again in March, the command becomes

```
> gcc -c March.c -o March.o  
> gcc strange_app.c March.o -o strange_app
```

Write a simple makefile to build the application.

Note that the .o file has to be re-generated only when the corresponding .c file changed, and the strange\_app binary has to be regenerated only when either the .o file or the strange\_app.c file changed.

- Stats:
  - Solved by: 62
  - Total submissions: 271
  - First to solve: bancel

Most of you did something like this:

```
MONTH=$(shell date +"%B")

strange_app: strange_app.o $(MONTH).o
    gcc strange_app.o $(MONTH).o -o strange_app

%.o: %.c
    gcc -c $< -o $@
```

Some put a different target to build each separate <month>.o :(

Most of you did something like this:

```
MONTH=$(shell date +"%B")

strange_app: strange_app.o $(MONTH).o
    gcc strange_app.o $(MONTH).o -o strange_app

%.o: %.c
    gcc -c $< -o $@
```

Some put a different target to build each separate <month>.o :(

But only one (kofman) really solved the challenge

```
MONTH=$(shell date '+%B')

strange_app: $(MONTH).o strange_app.o
    gcc strange_app.o $(MONTH).o -o strange_app
    @-ls *.o | grep -v $(MONTH).o | grep -v
strange_app.o | xargs rm
%.o : %.c
    $(CC) -c $(CFLAGS) $< -o $@
```



# Python 1

You want to write a python script to print the include tree of a certain C file. The code has to be able to extract the "#include" directive from the document passed as first parameter.

If the included files are included quotation (instead of angle-brackets), the python script has to recursively open the file and analyze the included statements. The final result is going to be a tree that you have to properly format and print.

For example, this is what the output of your program should look like:

```
> python solution.py code/fileA.c
code/fileA.c
|-- <stdlib.h>
|-- <stdio.h>
|-- header.h
```

- Solved by: ALL
- Total submissions: 154
- First to solve: bancel

## Solutions between 8 (!! ) and 59 lines of code



```
# rogdham

import sys
def cf(fn, p):
    with open(fn) as f:
        for l in [l.lstrip()[8:].strip() for l in f if l.lstrip()
[:8] == '#include']:
            print '%s-- %s' % (p, l.replace('"', ''))
            l[0] == '"' and cf(fn[:fn.rfind('/')] + '/' + l[1:-1],
p + ' |')
print sys.argv[1]
cf(sys.argv[1], ' |')
```

# Command-Line 2

CommandLineFu is a website in which, on a daily basis, users post useful shell commands. Other users can then vote on each command to express the fact that they find it useful, or down-vote it if they find it useless. The file "command\_line\_fu.example.html" in the current directory contains a dump of the webpage.

The goal of this assignment is to parse the example file and print only those commands with a number of votes  $\geq 5$  (the votes are expressed in the html in a `<div class="num-votes">` tag).

You can use any command-line tools you want for this assignment, but you cannot use python.

- Solved by: 62
- Total submissions: 128 (lower number)
- First to solve: pxthanh

```

</div>
  <div class="one-liner">
    <div class="line" title="Click to select this command">
      <div class="command">alias install='sudo apt-get install'</div>
    </div>
    <div style="display:none" class="sample-output" id="sample-output-9593">
      <div class="output-meta">
        This is sample output - yours may be different. </div>
      <div class="output">      <pre></pre>
.....
      <div class="votes">
      <div class="num-votes" id="num-votes-9593"> 7 </div>
      <div class="voting-options">

```

```
# lucab (75 characters)
```

```
awk -F' [<>] ' ' /"command"/ {cmd=$3}
      num-votes"/ {if ($3>=5) print cmd}' -
```



# Development 2

The goal of this assignment is to add one command to gdb named "fpointers" that locates pointers in a given area of memory.

The command (that has to be written in python) should take two parameters, the first representing the address of the memory region and the second its size. The output has to report all the pointers present in the area that point to other addresses in the same range.

- Stats:
  - Solved by: 51
  - Total submissions: 325
  - First to solve: haradwaith

```
import gdb, struct

class FPointers(gdb.Command):
    def __init__(self):
        gdb.Command.__init__(self, "fpointers", gdb.COMMAND_DATA)

    def invoke(self, arg, from_tty):
        prog = gdb.inferiors()[0]
        args = arg.split()
        start = int(args[0], 16)
        size = int(args[1])

        mem = prog.read_memory(start, size)
        for pos in range(len(mem)-8):
            x = struct.unpack("Q", mem[pos:pos+8])[0]
            if start <= x <= start+size:
                print "0x%x --> 0x%x"%(start+pos, x)
```

```
FPointers()
```

```
import gdb, struct

class FPointers(gdb.Command):
    def __init__(self):
        gdb.Command.__init__(self, "fpointers", gdb.COMMAND_DATA)

    def invoke(self, arg, from_tty):
        prog = gdb.inferiors()[0]
        args = arg.split()
        start = int(args[0], 16)
        size = int(args[1])

        mem = prog.read_memory(start, size)
        for pos in range(len(mem)-8):
            x = struct.unpack("Q", mem[pos:pos+8])[0]
            if start <= x <= start+size:
                print "0x%x --> 0x%x"%(start+pos, x)
```

FPointers()

Only 5 used read\_memory

Only 2 used struct

# Python 2

There is a strange server running on port 9999  
Please submit a python client that can talk to the server and solve  
all its puzzles.

- Stats:
  - Submissions: 348
  - Solved by: 59
  - First to solve: eeko

- 39% used line-based input
  - Last year was 16%

SERVER  
*(port 9999)*

CLIENT

Hello there... who are you?

foobar

Tell me your secret to start

xxxxx



You have to hardcode it  
You cannot read it from file

SERVER  
(port 9999)

CLIENT

Listen, I'm thinking about a number. Let's see if you can guess it

1050

Nope. Try something bigger

4500

Nope. Try something smaller

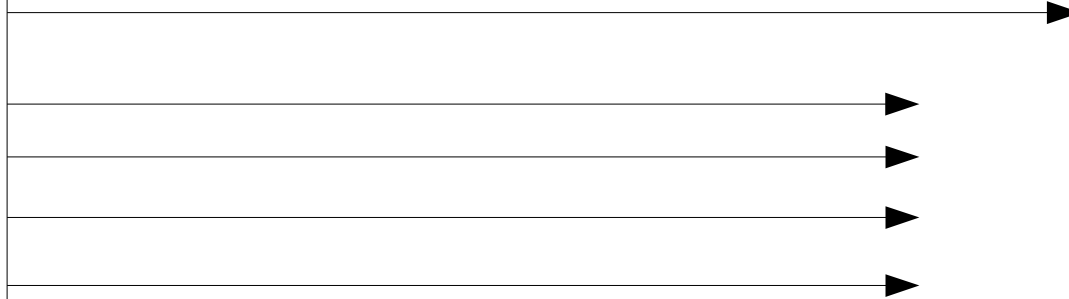
```
if buf == "nope. Try something bigger\n":  
    l = (l + r) / 2  
    s.send(str((l + r) / 2) + "\n")  
elif buf == "nope. Try something smaller\n":  
    r = (l + r) / 2  
    s.send(str((l + r) / 2) + "\n")
```

BRAVO :)

SERVER  
(port 9999)

CLIENT

Look at the next XX lines.



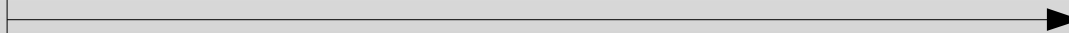
That was a pickle version of  
the current datetime.  
..tell me the number of  
microseconds...



```
f=client_socket.makefile()  
...  
microsec=(pickle.load(f)).microsecond
```



Legend -wait for it...- dary



SERVER  
(port 9999)

CLIENT

Listen, I was written on 12 15 2011.  
How many days ago was it?

```
date = re.match("Listen, I was written on (.*) . How many",line).group(1)  
res = (datetime.datetime.now() -  
       datetime.datetime.strptime(date,'%d %b %y')).days
```

Well Done  
My secret is XXXXX

# Command-Line 3

In order to save space on your new solid state disk, you want to automatically compress all the directories that do not contain at least one file used in the last month.

- Stats:
  - Solved by: 57
  - Total submissions: 283
  - First to solve: pxthanh
- All sort of solutions... from recursive bash functions (elegant) to single line!!

### # bidord (118 characters)

```
find -type d -exec sh -c 'if [[ -z $(find "{}" -type f -atime -30) ]];  
then tar czf "{}.tgz" "{}"; rm -r "{}"; fi' \;
```



Only 6 solutions without conditional statements

### # thainguyen (199 characters)

```
i=`find -type f -atime -30 | sed 's/\(.*\)\/[^/]*\/\1/' | sort -u`  
find -type d | sort -u | while read j; do  
    echo $i | grep -q "$j" || tar cfz "$j".tgz "$j"  
    echo $i | grep -q "$j" || rm -rf "$j"  
done
```

# Development 3

You are very proud of your new very lame program ("lame\_project.tgz") and you want to distribute it to the world so that it can be compiled for every existing system.

In order to do that you decided to use autotools.

For this challenge you have to write the required Makefile.am (maybe more than one) and the "configure.ac" and add them to the project.

The rules are simple:

- \* Lib\_foo must be compiled but NOT distributed // typo ... should have been installed

- \* The main application must be compiled and installed under  
`project/uselessbin/`

- \* The main application uses a "mystery\_value" function.

The function is defined in "mystery.h" but the actual library containing the implementation can be different in different systems. In particular, it can be implemented in one of these libraries: magic, voodoo, or mystery

Make sure that **\*\*the configure script\*\*** (and not yourself) can find the right one.

Moreover, the mystery\_value function can have one or two parameters depending on the systems. To be more general, main.h defines the appropriate "#define". Make sure that the configure script will set the variables according to the number of parameters.

```
# configure.ac

AC_SEARCH_LIBS([mystery_value], [voodoo mystery magic])

AC_COMPILE_IFELSE(
  [AC_LANG_PROGRAM([#include<mystery.h>], [mystery_value(2);]),
  [echo "One"
    AC_DEFINE([MISTERY_VALUE_ONEPARAM], [1],
              [The mystery value function has one parameter])],
  [echo "Two"])
```

```
# src/lib_foo/Makefile.am
noinst_LIBRARIES = libfoo.a
libfoo_a_SOURCES = foo.c foo.h
```

```
# src/main/Makefile.am
ubindir = ../../uselessbin
ubin_PROGRAMS = mystery_foo
mystery_foo_SOURCES = main.c main.h
mystery_foo_LDADD = ../lib_foo/libfoo.a
```

# Python 3

```

  V    V    VV   VV  V
#####O###
#####O##E
#####
#####
#####
#####
#####
S#####
#####

====>
  V    V    VV   VV  V
#####O###
#####*#*E
#####*#X#*#
#####*#*#*#*#
#####*#*#*#*#
#####*#*#*#*#
#####*#*#*#*#
S*#####
#####

```

- Solved by: 54
- Submissions: 361 (top number of submission)
- First to solve: bancel
- Shortest solution: 13 unreadable but very well commented LOC (rogdham)
- Longest solution: 377 LOC



Very smart.....

(...but try to read it)



```
import sys
with open(sys.argv[1]) as f:
    g=lambda p,d:p+{'U':-g.w, 'D':g.w, 'L':-1, 'R':1}[d]
    def z(queue):
        while queue:
            p,path=queue.pop()
            if z.m[p]=='E':
                return path
            z.m[p]='x'
            [queue.insert(0, (g(p,d),path+d)) for d in[d for d in'UDLR'if
z.m[g(p,d)]in'#E']]
            g.w,z.m=[(len(m[0]),[c for c in''.join(m+['O'for c in m[0]])])for m
in([l.strip()+ 'O'for l in f],)][0]
            [z.m.__setitem__(i,'O')for i in range(len(z.m)-1,g.w,-1)if z.m[i-
g.w]=='O']
            print z([(z.m.index('S'),'')])
```