Exercise 3

The application requires two arguments: (1) a username and (2) a password.

```
Use ./main <user> <password>
```

For example

```
> ./main roberto 1234
Start
Hello roberto
non authorized
End

> ./main roberto pwd0
Start
Hello roberto
authorized
End

> ./main hello pwd0
Start
Hello hello
authorized
End
```

Problem 3.1

Forge a username that makes the application to leak the password that is stored internally.

Since you probably need a username that contains "special" bytes, use the following procedure:

1. complete the python script `solution3.py`, which prints the forged output on the standard output
2. execute `./solution3.py > text` to generate a file that contains the forged username
3. execute `./main $(< text) hello`

The target `attack` of the Makefile automates tasks 2 and 3. Your solution consists of the script `solution3.py`.
To test your solution execute =./test.py= or =py.test test.py=.

Hints

Debug the program using GDB. Find the distance between the location of the variable `name` and `saved rip`.
Do not worry if the program crashes after leaking the password.