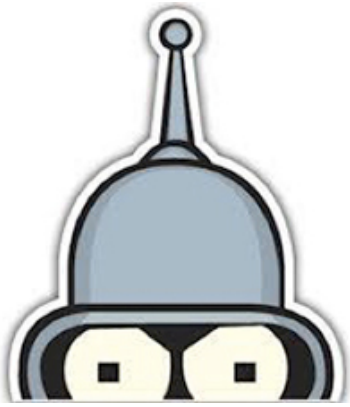
B2 - Stumpers

B-CPE-210

Fridge

Duo Stumper



1.06

{EPITECH.}

Fridge

binary name:	fridge
repository name:	CPE_duostumper_$STUMPERNUMBER_$ACADEMICYEAR
repository rights:	ramassage-tek
language:	C
compilation:	via Makefile, including re, clean and fclean rules

- Your repository must contain the totality of your source files, but no useless files (binary, temp files, obj files,...).
- Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).

The $STUMPERNUMBER is always on one digit: `1`, `2`, ...

For this project, the **only** authorized functions are those of the standard `libC`.

The goal of your program is to reproduce the functioning of a refrigerator in C language. This refrigerator will contain the ingredients needed to make pizza and pasta.

The first thing to do is to create the refrigerator and put the ingredients inside.
A `"disp fridge"` command should allow you to display fridge's ingredients.
Another command, `"addToFridge"` will allow you to add ingredients.
The ingredients are as follows:

- tomato
- dough
- onion
- pasta
- olive
- pepper
- ham
- cheese

The `"disp fridge"` command should display the ingredients in the following way:

```
ingredient1 = quantity1
ingredient2 = quantity2
...
```

At the program's startup, the fridge should contain 50 of each ingredient.)

Italic yellow text represents user input.

Terminal

```
~/B-CPE-210> ./fridge
poop
'poop': Invalid operation
disp fridge
tomato = 50
dough = 50
onion = 50
pasta = 50
olive = 50
pepper = 50
ham = 50
cheese = 50
addToFridge olive 50
disp fridge
tomato = 50
dough = 50
onion = 50
pasta = 50
olive = 100
pepper = 50
ham = 50
cheese = 50
```

In the previous example, the program was exited by sending an End-Of-File via Ctrl+D.

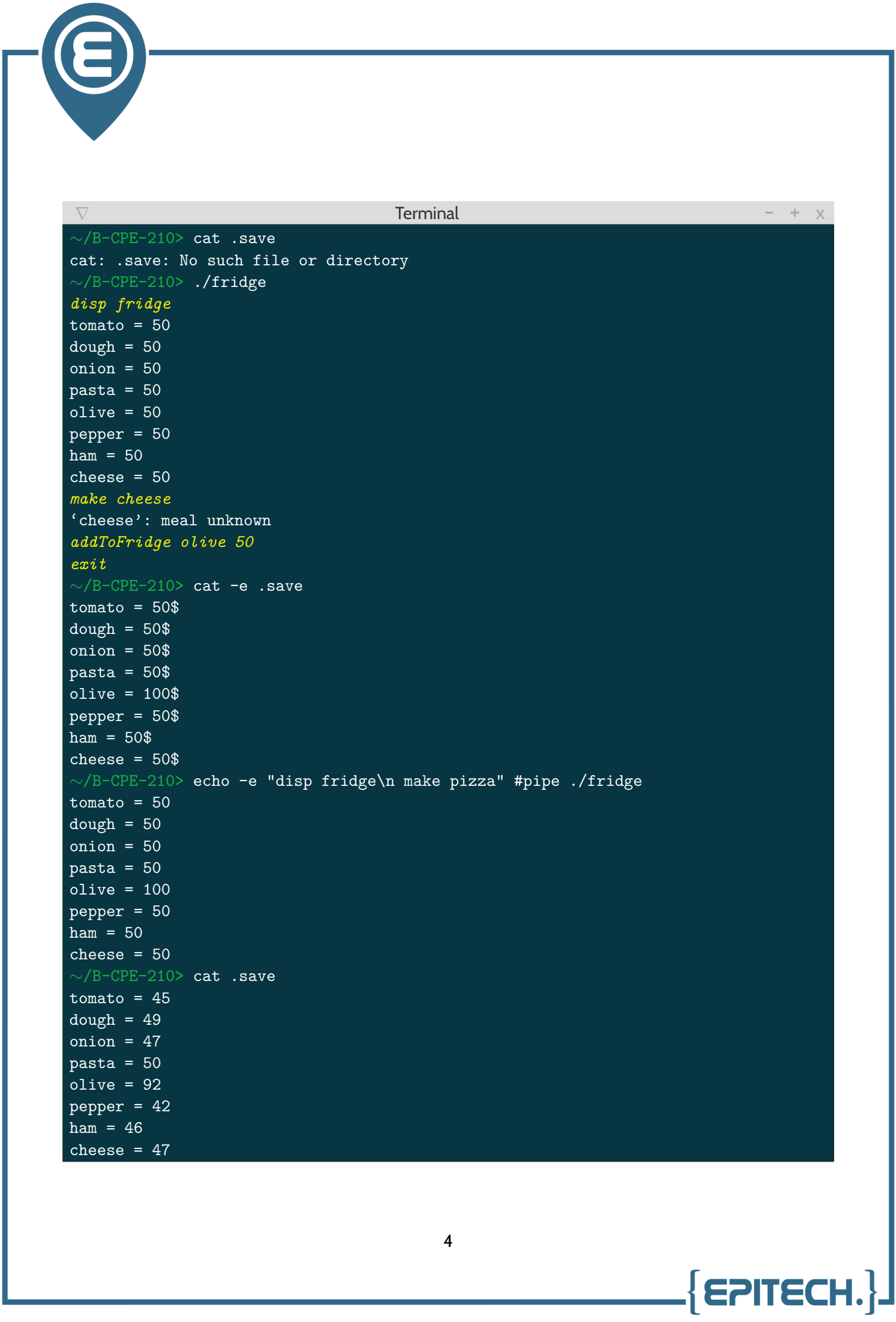
Next, the user should be be able to make something to eat (because it's vital).
In order to do this, you should implement a `"make"` command. This will take an argument as parameter that indicates the meal to be made:
- **"make pizza"** will prepare a pizza from 5 tomatoes, 1 portion of pizza dough, 3 onions, 8 olives, 8 peppers, 4 portions of ham and 3 portions of cheese. (Of course, the ingredients are to be taken out of the stock.)
- **"make pasta"** will prepare a plate of pasta from 5 tomatoes, 2 portions of pasta, 6 olives, 3 portions of cheese and 4 portions of ham.

An "exit" command will also allow you to exit the program gracefully.
If the fridge doesn't have enough ingredients to make the meal, an error message will be displayed (on standard output).
If the command is invalid an "Invalid operation" message will be displayed (on standard output).
Now, you need a way to save the refrigerator's state. When you exit the program, the refrigerator's contents

must be written in a ".save" file that will be read again when the program starts up in order to retrieve the previous contents. If the save file is invalid, it is an error, you must exit accordingly. If it doesn't exist, it must be created and initialized with the default values.

Terminal

```
~/B-CPE-210> cat .save
tomato = 50
dough = 50
onion = 50
pasta = 50
olive = 50
pepper = 3
ham = 50
cheese = 50
~/B-CPE-210> ./fridge
make pizza
'make pizza': not enough pepper
disp
'disp': Invalid operation
addToFridge horse 50
'addToFridge horse 50': Invalid operation
```


Terminal
~/B-CPE-210> cat .save
cat: .save: No such file or directory
~/B-CPE-210> ./fridge
disp fridge
tomato = 50
dough = 50
onion = 50
pasta = 50
olive = 50
pepper = 50
ham = 50
cheese = 50
make cheese
‘cheese’: meal unknown
addToFridge olive 50
exit
~/B-CPE-210> cat -e .save
tomato = 50$
dough = 50$
onion = 50$
pasta = 50$
olive = 100$
pepper = 50$
ham = 50$
cheese = 50$
~/B-CPE-210> echo -e "disp fridge\n make pizza" #pipe ./fridge
tomato = 50
dough = 50
onion = 50
pasta = 50
olive = 100
pepper = 50
ham = 50
cheese = 50
~/B-CPE-210> cat .save
tomato = 45
dough = 49
onion = 47
pasta = 50
olive = 92
pepper = 42
ham = 46
cheese = 47