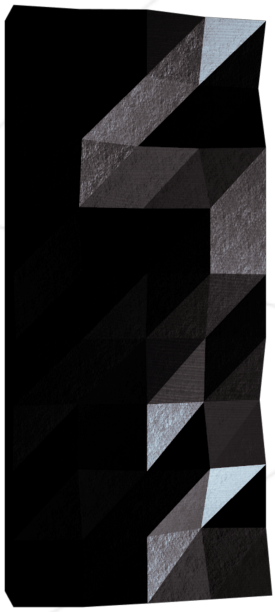

C Piscine

Rush 02

Summary: This document is the subject for Rush02 of the C Piscine @ 42.

Version: 10.5

Contents

Chapter I

Foreword

Here is an old-fashioned pecan pie recipe for you:

```
Ingredients
- Pastry dough
- 3/4 stick unsalted butter
- 1 1/4 cups packed light brown sugar
- 3/4 cup light corn syrup
- 2 teaspoons pure vanilla extract
- 1/2 teaspoon grated orange zest
- 1/4 teaspoon salt
- 3 large eggs
- 2 cups pecan halves (1/2 pound)
Accompaniment: whipped cream or vanilla ice cream

Preparation
Preheat the oven to 350°F with a baking sheet on middle rack.
Roll out the dough on a lightly floured surface with a lightly floured rolling pin
into a 12 inch round and fit it into a 9 inch pie plate.
Trim the edge, leaving a 1/2-inch overhang.
Fold the overhang under and lightly press it against rim of pie plate, then crimp
decoratively.
Lightly prick the bottom all over with a fork.
Chill until firm, at least 30 minutes (or freeze for 10 minutes).
Meanwhile, melt the butter in a small heavy saucepan over medium heat.
Add the brown sugar, whisking until smooth.
Remove from the heat and whisk in corn syrup, vanilla, zest, and salt.
Lightly beat the eggs in a medium bowl, then whisk in corn syrup mixture.
Put the pecans in pie shell and pour corn syrup mixture evenly over them.
Bake on the hot baking sheet until filling is set, 50 minutes to 1 hour.
Let it cool completely.

Cook's notes:
The pie can be baked 1 day ahead and chilled. Bring it to room temperature before
serving.
```

Chapter II

Instructions

- You must do the project with the imposed team. Meeting with your teammates is part of your tasks, using any appropriate means (Slack, email, phone if available, or directly in person).
- If you have really tried everything to reach one of your teammates, but they still remain unreachable, follow the instructions from your pedagogical team, if provided. The default procedure is to do the project with the available teammates, and discuss the situation during the evaluation. Even if the group leader is missing, you still have access to the submission directory.
- Your work must comply with the Norm. If you have bonus files/functions, they are included in the norm check, and you will receive a score of 0 if there is a norm error.
- You must handle errors coherently. You may either print an error message or simply return control to the user.
- Your project must be completed and pushed to the Git repository by the deadline displayed on the project's main page on the intranet.
- You must follow the submission procedure described at the end of this document, if provided.
- Your program must compile using `cc` with the following flags: -Wall -Wextra -Werror. If there is a moulinette, it will use the same compiler and flags.
- If your program does not compile, you will receive a score of `0`.
- The group will be automatically registered for the defense. You must attend your evaluation with all of your teammates. The purpose of the defense is to present and explain your work comprehensively.
- Do not cancel your evaluation; you will not get a second one.
- Each member of the group must be fully aware of all the details of the project. If you choose to split the workload, ensure you understand every part completed by other team members. This understanding may be verified during the evaluation.
- You must submit a Makefile, which will compile your project using the rules `$NAME`, `clean` and `fclean`.

Chapter III

Subject

	Exercise 00
rush-02	
Turn-in directory: *ex*00/	
Files to turn in: `Makefile and all the necessary files`	
Allowed functions: `write, malloc, free, open, read, close`	

- Create a program that takes a number as an argument and converts it to its written letter value.
- Executable name: `rush-02`
- Your source code will be compiled as follows:

```
make fclean
make
```

- Your program can take up to 2 arguments:
  - If there is only one argument, it is the number that you need to convert.
  - If there are two arguments, the first argument is the new reference dictionary and the second argument is the number that you need to convert.
- If the argument representing the number is not a valid and positive integer, your program must output "Error" followed by a newline.

`Your program should handle numbers beyond the range of unsigned int.`

- Your program must parse the dictionary given as a resource to the project. The values inside it must be used to print the result. These values can be modified.
- Any memory allocated on the heap (with malloc(3)) must be freed correctly. This will be verified during the evaluation.
- The dictionary will have the following rules:

```
[a number][0 to n spaces]:[0 to n spaces][any printable characters]\n
```

  - You will trim the spaces before and after the values in the dictionary.
  - The dictionary will always have at least the keys contained in the reference dictionary. Their values can be modified, more entries can be added, but the initial keys can't be removed.
  - The entries of the dictionary can be stored in any order.
  - There can be empty lines in the dictionary.
  - If you have any errors from the dictionary parsing, your program must output "Dict Error\n".
  - If the dictionary does not allow you to perform the conversion of the provided number, your program must output "Dict Error\n".

- Example:

```
$> ./rush-02 42 | cat -e
forty two$
$> ./rush-02 0 | cat -e
zero$
$> ./rush-02 10.4 | cat -e
Error$
$> ./rush-02 100000 | cat -e
one hundred thousand$
$> grep "20" numbers.dict | cat -e
20    :     hey everybody ! $
$> ./rush-02 20 | cat -e
hey everybody !$
```

Chapter IV

Bonuses

You can complete the following bonuses:

- Using "`-`", "`,`", "`and`" to be closer to the correct written syntax.
- Doing the same exercise in a different language, using another dictionary which will contain the translated entries.
- Using `read` to read numbers from the standard input (one per line) when the command line argument for the number is "`-`".
  Example:

```
$> ./rush-02 -
42
forty two
0
zero
^D
$>
```

Chapter V

Submission and peer-evaluation

Submit your assignment to your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Do not hesitate to double-check the names of your files to ensure they are correct.

This assignment is not verified by a program. You are free to organize your files as you wish, provided you submit the mandatory files and comply with the requirements.

`Submit only the files requested by the subject of this project.`