main

command

exit

Arrête le program

```
    elif command == "exit":
        print("exiting the program")
        break
```

Autres commandes

help

Liste toutes les commandes possible et leur fonctions:

```
help = '''commands:
        - test: executes a test
        - list: choose a folder and see whats inside of it
        - edit: gives you the ability to manipulate text files.
        - occ: lets you see how many times a letter is occurring in the chosen text file
        - exit: stops the program
        '''
```

main

list

edit

Demande du nom du répertoire a lister les fichier dedans:

```
direct_name = input("Please input directory name that you need to list:")
```

```
try:
    directory = os.listdir("folders/"+direct_name)
    return(directory)
#when the folder isn't found, we ask again
except FileNotFoundError:
    print("directory not found")
```

Si on le trouve : on liste les fichiers dans le repertoire choisi

Sinon: on dit que le répertoire n’est pas trouver et on redemande

main

edit

CHECKER():

```
def checker():
    while True:
        print(os.listdir("folders"))
        direct_name = input("Please input directory name: ")
        #the test variable is used to test for the existence of a directory
        test = os.path.exists("folders/"+direct_name)
        # if dir exists now ask for file
        if test:
            lister= os.listdir("folders/"+direct_name)
            #checks if the directory is empty because there wont be any files to edit
            if lister == []:
                print("directory empty")
                break
            #lists alll the files in that dir to help the user
            print(lister)
            #and if the dir isnt empty we will ask for the file inside of it
            while True:
                file_name = input("Please input file name(without file extention): ")
                #checks if the file exists
                test = os.path.exists("folders/"+direct_name+"/"+file_name+".txt")
                if test:
                    #we add the directory name and file name to make a complete path and return it to the edit function
                    #to open the chosen file
                    path = "folders/"+direct_name+"/"+file_name+".txt"
                    return(path)
                else:#if the file isnt found, we ask again
                    print("File not found")
        else:
            #if directory isnt found, we ask again
            print("Directory not found")
```

On demande le répertoire et voit si elle existe sinon, on redemande. Ensuite on demande le fichiers qui est dans la liste et si il existe on retourne au variable “path” la route du fichier:

```
path = checker()
```

ACTION():

```
def action():
    help = '''commands:
        - read: read the contents of the chosen file
        - overwrite: empty a file and write to it
        - write: add o a file
        '''
    while True:
        #read - r
        #overwrite - w+
        #write - a+
        action_input = input("how would you like to edit it?: ")
        if action_input == "read":
            return "r"
        elif action_input == "overwrite":
            return "w+"
        elif action_input == "write":
            return "a+"
        elif action_input == "help":
            print(help)
        else:
            print("action unavailable, command 'help' is available")
```

Puis on demande se qu’on veut faire avec le fichier choisi, on a 3 modes:

#read - r

#overwrite - w+

#write - a+

Et si on demande le mauvaise action on la commande help marchera ici.

Et on fini par donne l’action au variable : `act = action()`

EDIT():

```
def edit():
    #we use 2 fuinction to get the variables needed to open a file
    path = checker()
    act = action()
    file = open(path, act)
    # we just need to read we wont be writing to the file
    if act == "r":
        print(file.read())
    else:
        data = input("what would you like to write: ")
        file.write(data)
    file.close()
```

Avec la route du fichier et l'action qu'on va faire, on ouvre le fichier:

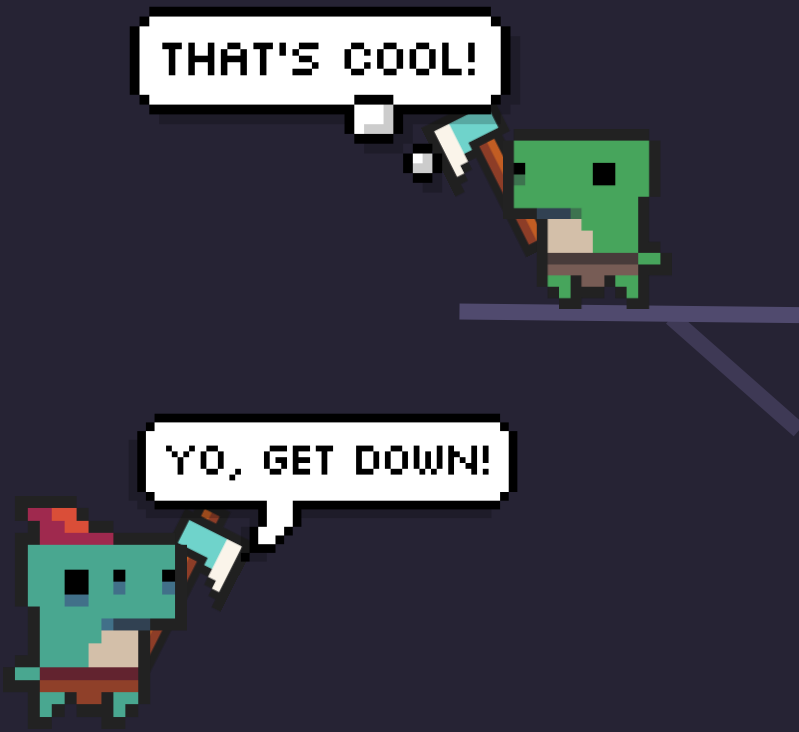
```
file = open(path, act)
```

Ensuite en vois si on veut que lire le fichier, on le lit:

```
if act == "r":
    print(file.read())
else:
    data = input("what would you like to write: ")
    file.write(data)
file.close()
```

Sinon on demande de se que on veut écrire au fichiers et on 'write' ou 'overwrite' par rapport a l'action choisit.

Et enfin on ferme le fichier.



main

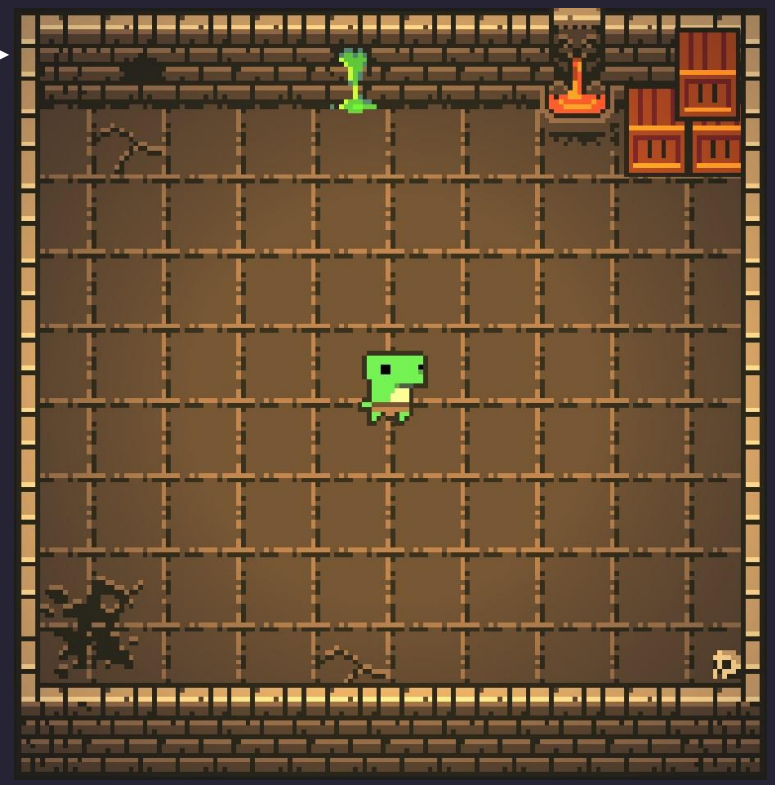
occ

```
alph_dict = {"a": 0,
             "b": 0,
             "c": 0,
             "d": 0,
             "e": 0,
             "f": 0,
             "g": 0,
             "h": 0,
             "i": 0,
             "j": 0,
             "k": 0,
             "l": 0,
             "m": 0,
             "n": 0,
             "o": 0,
             "p": 0,
             "q": 0,
             "r": 0,
             "s": 0,
             "t": 0,
             "u": 0,
             "v": 0,
             "w": 0,
             "x": 0,
             "y": 0,
             "z": 0,
             }
```

OCC():

```
def occ():
    file_loc = checker()
    file = open(file_loc, 'r')
    lenght = 0
    while True:
        # read by character
        char = file.read(1).lower()
        if char in alph_dict:
            alph_dict[char] += 1
            lenght += 1
        if not char:
            break
    file.close()
    drawer(lenght)
```

On commence par passe dans chaque lettre du fichier et si la lettre existe dans le dictionnaire :
on l'ajoute a la key qui représente cette lettre
Et enfin quand il y a plus de lettre on arrête la tant que loop



Il y avait pas assez de place pour la fonction suivante :

Réutilisation de la fonction 'checker()' pour demander la location du fichier

DRAWER():

```
def drawer(lenght):
    # we take the keys make turn the into name to put on the horizontal axis
    names = list(alph_dict.keys())
    # and their values into a list so we can turn them into %
    # values_deci is the list with the values in deciaml
    values_deci = list(alph_dict.values())
    #we transform the decimal list into % using the fonction percent
    values = percent(values_deci, lenght)
    plt.bar(range(len(alph_dict)), values, tick_label=names)
    # we update the dict for the percent to visualize it without the graph
    pers_dict =  dict_updater(values)
    print(pers_dict)
    plt.show()
```

On commence par placer les 'keys' dans le dictionnaire sur le axe horizontal et ensuite pour l'axe vertical on prend les nombres obtenu du scan qu'on a fait et on les envoi vers la **fonction de %** pour les transformer en % par rapport a la longueur tu fichier texte

PERCENT():

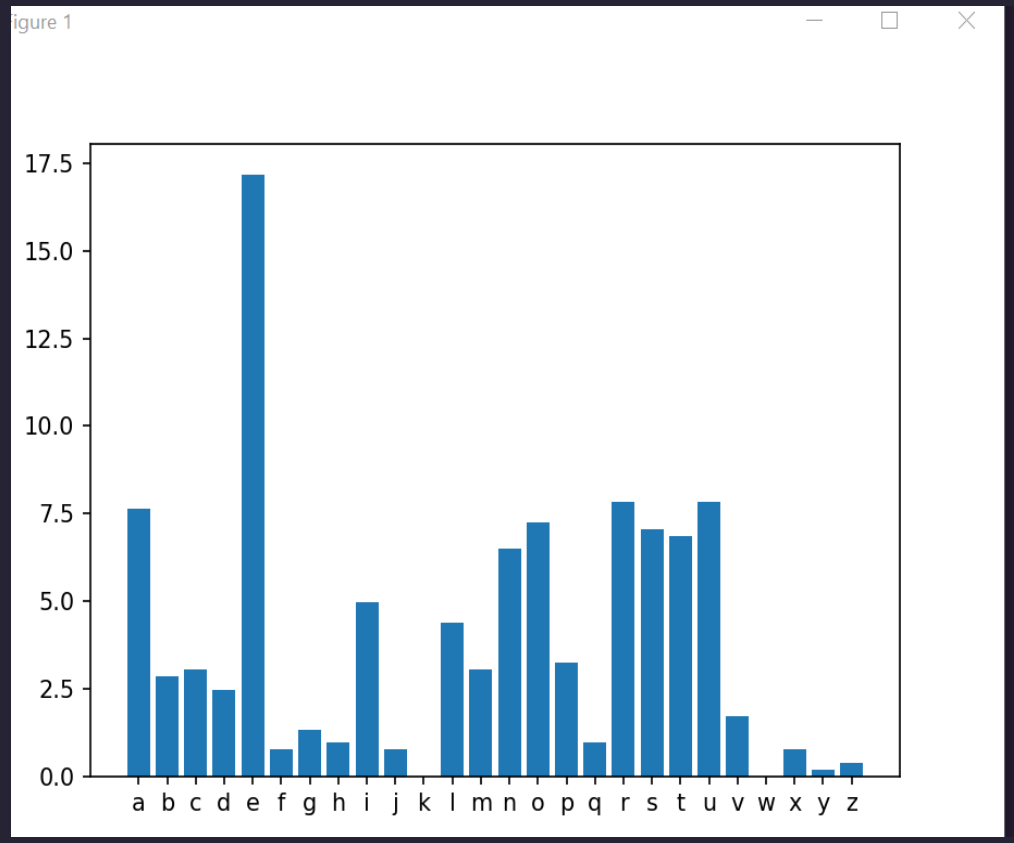
```
def percent(values_deci, lenght):
    #values_pers is the list that will be used in the graph
    values_pers = []
    # we transform the values into %
    for item in values_deci:
        val = round((item/lenght)*100, 3)
        values_pers.append(val)
    return values_pers
```

Ici on prend la liste des valeurs décimal et on passe par chaque nombre dans la liste en le transformant en nombre % et le donnant a une liste qui va être le variable 'valeurs'

DICT_UPDATE():

```
def dict_updater(uptListe):
    #the keys used in the dicts
    alphabet = "abcdefghikjlmnopqrstuvwxyz"
    for i in range (len(alphabet)):
        # we passe by each key using the string "alphabet"
        # to update them using the list created with the percent fonction
        pers_dict[alphabet[i]] = str(uptListe[i])+"%"
    return pers_dict
```

Utilisant les 'keys' dans le variable alphabet, on place les % dans un **dictionnaire(page suivante)** pour que l'utilisateur puisse comparer plus facilement ces textes



EXAMPLE

```
) occ
['test', 'texts']
Please input directory name: texts
['file1.txt', 'file2.txt', 'file3.txt', 'file4.txt', 'file5.txt']
Please input file name(without file extention): file1
{'a': '7.634%', 'b': '2.863%', 'c': '3.053%', 'd': '2.481%', 'e': '17.176%'
, 'f': '0.763%', 'g': '1.336%', 'h': '0.954%', 'i': '4.962%', 'j': '0.0%',
'k': '0.763%', 'l': '4.389%', 'm': '3.053%', 'n': '6.489%', 'o': '7.252%',
'p': '3.244%', 'q': '0.954%', 'r': '7.824%', 's': '7.061%', 't': '6.87%', '
u': '7.824%', 'v': '1.718%', 'w': '0.0%', 'x': '0.763%', 'y': '0.191%', 'z'
: '0.382%'}
```

CMD

```
pers_dict = {"a": 0,
             "b": 0,
             "c": 0,
             "d": 0,
             "e": 0,
             "f": 0,
             "g": 0,
             "h": 0,
             "i": 0,
             "j": 0,
             "k": 0,
             "l": 0,
             "m": 0,
             "n": 0,
             "o": 0,
             "p": 0,
             "q": 0,
             "r": 0,
             "s": 0,
             "t": 0,
             "u": 0,
             "v": 0,
             "w": 0,
             "x": 0,
             "y": 0,
             "z": 0,
             }
```