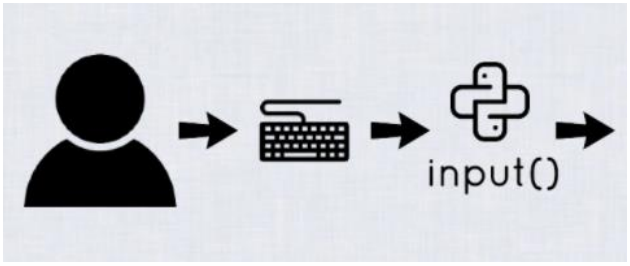
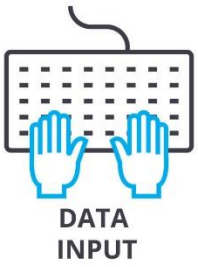
External Files (persistent data)

Storing/getting data from outside our programs



So far, we have mainly been using simple (**user generated**) text, numbers, and mouse clicks as **input** for our programs. Most computer programs are designed to interact with humans, so this type of **input** is very important to know about.

The **data types** we have been using to **store** information: *variables*, *lists*, *dictionaries* are referred to as **temporary data storage.** Their values can get changed or deleted each time we run our program.



Another very important **data type** is **Files**.

Files are a **persistent** data type that *allow us to store information* **outside of our programs**. Files are an incredibly common and important data type that allow us to get input from other programs, computers, the internet, or secondary storage devices.

Common types of files:

a) Python views **text files (.txt)** as a series of characters.
b) Python can view images, videos, and more as a **binary file (.bin)** (or a series of bytes).
c) **CSV files:** A CSV file is **a Comma Separated Values** file. This is a plain text file that uses specific to arrange data. It's great for spreadsheets and data bases, but is used for all sorts of data transfer. **(.csv)**

File Handling in Python

1. Opening file
2. Read or write (perform operation) with files
3. Close the file

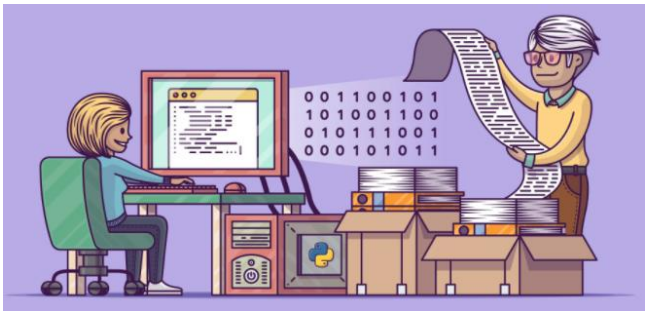
Text files (.txt):

Text files are files that have a record of a string of characters. These files are unformatted but do **recognise when text goes on to a separate line**. Useful data in the form of **.txt** files can be found all over the internet, on your computer, or make your own. Try the following sites to find some fun text files you can use for the following exercises:

https://filesamples.com/formats/txt

Here you can generate your own.

https://textdoc.co/




```
file_name = input("Enter file name: ")
file = open(file_name)
contents = file.read()
print (contents)
```

Exercise#1 (opening a text file)

Create or find a text file using the web address show *before* the code. Get the .txt file into the download folder of your laptop. Now, **upload** a text file into Replit *Then* use the code above to **open** and print the contest of the text file.

In this exercise please note: you must **open** a file in python before you do anything with it!

Important info about "opening" files in python!

The **open** command used for open files in python has several important *modes*. Each mode had to do with what you want to do with the file.

1. **Read** the file – means open it up and have it contents available to view or be printed.
2. **Write** the file – allows you to create an empty file or adds an existing file by **deleting** and writing over the first file (starting with the beginning of the file)
3. **Append** – add to the *end* of a file with messing with the rest of it.

```
file = open(file_name)       # read only
file = open(file_name,"r")   # read only
file = open(file_name,"w")   # creates a new file or deletes old file
file = open(file_name,"a")   # opens and allow you to add to the end of the file
```

1. **Rea**d Only (**'r'**) : Open text file for **reading**. Curser is positioned at the beginning of the file.

2. **Write** Only ('**w**') : Open the file for **writing**. For the existing files, the data is **deleted** and over-written. The curser is positioned at the beginning of the file. Creates the file if the file does not exist.

3. **Append Only** ('**a'**): Open the file for **writing**. The file is created if it does not exist. The data being written will be **inserted at the end, after the existing data**.

Exercise#2 Let the user create a file.

```
text=input('type a sentence to be stored: ')     # user inputs data
file = open('output.txt', 'w')                   # create/write to file -notice the 'w'
file.write(text)                                 # Write user input to file
file.close()                                     # Close file (for safety)
contents=open('output.txt', 'r')                 # put file data into a variable
contents=contents.read()                         # make file a readable string
print(contents)                                  # print the string
```

Copy the code above into Replit make sure you read the comments. Run the code and see if a **output.txt** file is created in the files section of your Replit. Run the program repeatedly and notice how the data the .txt file will change each time you run the program. Remember with files, you *can* leave your Replit and most recent file will be as it was when you left it.

Exercise#3

WARNING: The nastiness of writing to a file

Add each block of code below **separately** (on it's own) to *the end of the code in **exercise#2***. Carefully watch what each code does and make sure to check the contents of your .txt file in the files section of your Replit.

```
contents=open('output.txt', 'w')
contents.write('Your file is gone! Now it is just this')
contents=open('output.txt', 'r')
contents=contents.read()
print(contents)
```

```
contents=open('output.txt', 'w')
contents=open('output.txt', 'r')
contents=contents.read()
print(contents)
```

Exercise#4

Create a text file of a list of 10 first names and high scores on the Pokémon GO video game upload the file to Replit, **open it, and print it to the screen**. You can create your .txt file in Microsoft Notepad (or however you'd like)

Example:

```
Jim 34
Keren 23
Billy 45
Nigel 120
Cindy 145
.
.
.
```

Exercise#5 Appending to a text file

Enter the following code into a replit and upload the .txt file you created in exercise#3. Examine the code carefully so you can see know how the **'a' – append** mode works when you open a file. Run the code and add to the file. Check the .txt file in the file section of your Replit to see if it's updating as you would expect.

```
newstuff=input("Add a name and high score: ") # ask for info to ADD to file
file = open(file_name,"a")                     # open file so you can "append" info
file.write("\n")                               # put the cursor on a new line
file.write(newstuff)                           # add new stuff to the file
file.close()                                   # close file (as good practice)
file = open(file_name)                         # open file for reading
contents = file.read()                        # put file's contents into a variable
print (contents)                               # print the contents of the file
```

Exercise#6

Imagine you are a working at a coffee shop company that has 3 stores in Whistler village. First, create 3 .txt files that have 5 employee names each and upload them to Replit. Now, create a program that can ask a user for what file they would like to select (you should already uploaded to Replit). Now give the user some options. Allow them to see the list of employees in that file and/or add to that file.

Exercise#7

Your boss at an engineering firm wants you to monitor *thousands* of temperature sensors on a newly designed bridge that has heating elements in the asphalt to keep the surface from freezing. Each sensor records the temperature at a different spot on the bridge. During testing it appears something is wrong with the heating elements and your boss wants you to continually count the number of times that any sensors reads **below zero** degrees Celsius. See next page for further instructions…

Exercise#7 continued….

1. Go https://www.flyrnai.org/compleat/ExampleFiles.jsp
2. download the FlyRNAi_data_baseline.txt file.
3. Save and upload the file to a new python Replit.
4. Create a program that does the following:

    a) Open the file for reading
    b) Print the file
    c) Count the number of negative signs in the file '-'
    d) Report the result to the user.

There is a partial solution below, but please **try to do it on your own first**!!!!

Hey!

Possible Solution:

```
with open ("temp data.txt", "r") as myfile:
    data = myfile.read()
    freq = 0
    for x in data:
        if x =='-':
            freq = freq + 1
    print(freq)
```

Exercise#8

Create a program that will count the **total number of sensor readings** in the previous .txt file.

Exercise#9

Use Python to create a dice rolling game that will *store results of each game in a **file***.
Your dice game should:

1. Ask for two players names
2. Ask the first user to press a key to roll the dice (by generating a random number between 1 and 12)
3. Ask the second user to press a key to roll the dice.
4. Report each dice roll to the screen
5. Report the winner of the game (who ever has the higher roll).
6. After each game store the winners name in a file called **`winners.txt`**
7. Create an option in your game that will report the contents of **`winners.txt`** at anytime.