ISLab Python Course

Session 7: Working with Files

Presenters:

Shahrzad Shashaani

Hamed Homaei Rad

Saeed Samimi

Summer 2023

K.N.Toosi University of Technology

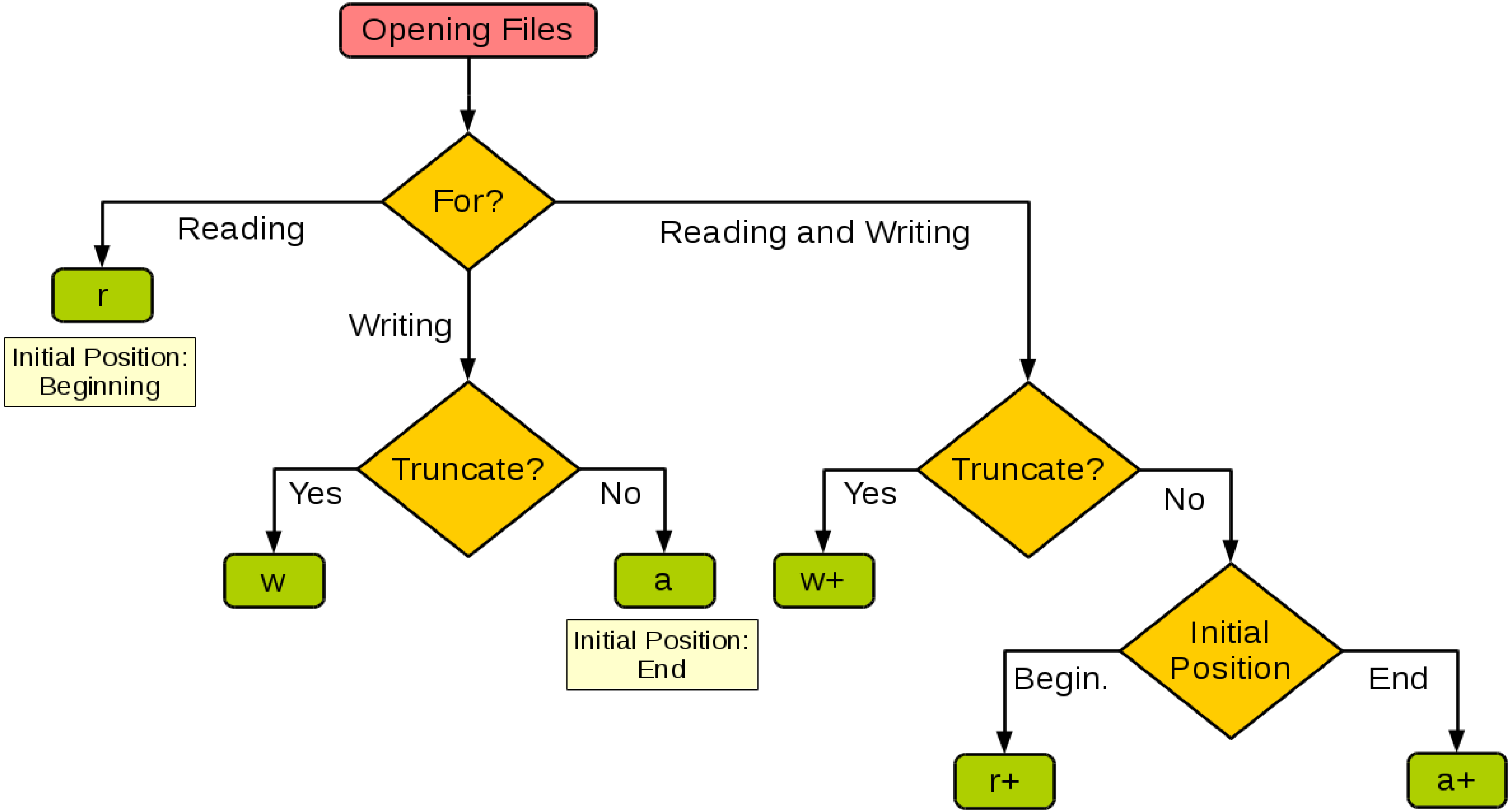
File Handling Modes

- File handling modes in Python determine how a file should be opened and how we can make changes in the file
- **'r' (Read Mode)**
  - It opens the file for reading
  - It is the default mode
- **'w' (Write Mode)**
  - It opens the file for writing
  - If the file already exists, it's emptied. If it doesn't exist, a new file is created
- **'a' (Append Mode)**
  - It opens the file for writing and it won’t overwrite existing data

File Handling Modes

- **'b' (Binary Mode)**
  - It is to indicate that the file should be treated as binary
  - It is used for working with non-text files, like images or audio
- **'x' (Exclusive Creation Mode)**
  - It is used to create a new file
- **'r+' (Read and Write Mode)**
  - With this mode the file is opened for both reading and writing
- **'w+' (Write and Read Mode)**
  - This mode allows both reading and writing. However, it empties the file if it exists and creates a new one if it doesn’t
- **‘a+’ (Append and Read Mode)**
  - It opens the file for reading and writing and won’t overwrite existing data

File Handling Important Modes



https://stackoverflow.com/questions/1466000/difference-between-modes-a-a-w-w-and-r-in-built-in-open-function

Opening and Reading Files

- **open()**

```
file = open(file_path, mode)
```

- **read()**

```
file = open(file_path, 'r')
content = file.read()
```

- **readline()**

```
file = open(file_path, 'r')
line = file.readline()
```

- **readlines()**

```
file = open(file_path, 'r')
lines = file.readlines()
```

Writing Files

- **open()**

```
file = open(file_path, mode)
```

- **write()**

```
file = open(file_path, 'w')
file.write('Hello World\n')
```

seek() Function

- **seek()**

```
file = open(file_path, mode)
file.seek(offset[, whence])
```

- **offset**
  - Number of positions to move forward
- **whence**
  - It defines the point of reference
    - 0: Sets the reference point at the beginning of the file **(default value)**
    - 1: Sets the reference point at the current file position **(file must be opened in binary mode)**
    - 2: Sets the reference point at the end of the file **(file must be opened in binary mode)**

Handling File Exceptions

- **Try-except**

```
try:
        file = open(file_path, 'r'):
        content = file.read()
except FileNotFoundError:
        print(f"The file '{file_path}' does not exist.")
except PermissionError:
        print(f"You don't have permission to access '{file_path}'.")
except Exception as e:
        print(f"An error occurred: {e}")
```

Closing Function

- **close()**
  - is used to close a file or a network connection that has been previously opened
  - If you don't close it properly, those resources might not be released until the program exits, potentially leading to resource leaks and decreased performance.

```
file = open(file_path, 'r’)
content = file.read()
file.close()
```

The Usage of ‘with’ Statement

- It ensures that the resources are appropriately acquired and released, even in the presence of exceptions
- It is helpful for where you need to perform some setup before using a resource and clean up afterward

```
with expression as variable:
```

```
with open(file_path, ‘r’) as file:
        contents = file.read()
        print(contents)
```

The Usage of ‘with’ Statement

- **Automatic Resource Management**
  - The resource (in this case, the file) is automatically released or closed
- **Exception Safety**
  - If an exception occurs, the resource is still released properly
- **Eliminates the Need for Manual Cleanup**
  - It doesn’t need to use ‘close()’ function

Pickle Module in Python

- Offers a way to turn Python objects into a compact form that can be stored or transmitted, and then restore them to their original state
- Converting a Python object into a byte stream
- It can be used to store and retrieve all structures supported by Python
    - Numbers (int, float, complex)
    - Strings (str, bytes)
    - Tuples, lists, Sets and dictionaries
    - Defined classes and objects
    - Functions and lambda expressions
    - Exceptions
- It can be used to transfer data between Python programs or between different programming languages that support the pickle format