SNS COLLEGE OF ENGINEERING
Kurumbapalayam (Po), Coimbatore – 641 107

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

19IT103 – COMPUTATIONAL THINKING AND PYTHON PROGRAMMING

- **A readable, dynamic, pleasant, flexible, fast and powerful language**

Objective

Files and exception: text files, reading and writing files, format operator; command line arguments, errors and exceptions, handling exceptions, modules, packages; Illustrative programs: word count, copy file, Voter's age validation, Marks range validation (0-100).

File

A file is a container in a computer system for storing information. Files used in computers are similar in features to that of paper documents used in library and office files. In a computer operating system, files can be stored on optical drives, hard drives or other types of storage devices.

Exceptions

- **Exception: error that occurs while a program is running**
  - Usually causes program to abruptly halt
- **Traceback: error message that gives information regarding line numbers that caused the exception**
  - Indicates the type of exception and brief description of the error that caused exception to be raised

Exceptions (cont'd.)

- **Many exceptions can be prevented by careful coding**
  - Example: input validation
  - Usually involve a simple decision construct
- **Some exceptions cannot be avoided by careful coding**
  - Examples
    - Trying to convert non-numeric string to an integer
    - Trying to open for reading a file that doesn't exist

Exceptions (cont'd.)

- **Exception handler: code that responds when exceptions are raised and prevents program from crashing**
  - In Python, written as `try/except` statement
    - General format:

```
try:
    statements
except exceptionName:
    statements
```

    - Try suite: statements that can potentially raise an exception
    - Handler: statements contained in `except` block

Exceptions (cont'd.)

- **If statement in try suite raises exception:**
  - Exception specified in except clause:
    - Handler immediately following except clause executes
    - Continue program after try/except statement
  - Other exceptions:
    - Program halts with traceback error message
- **If no exception is raised, handlers are skipped**

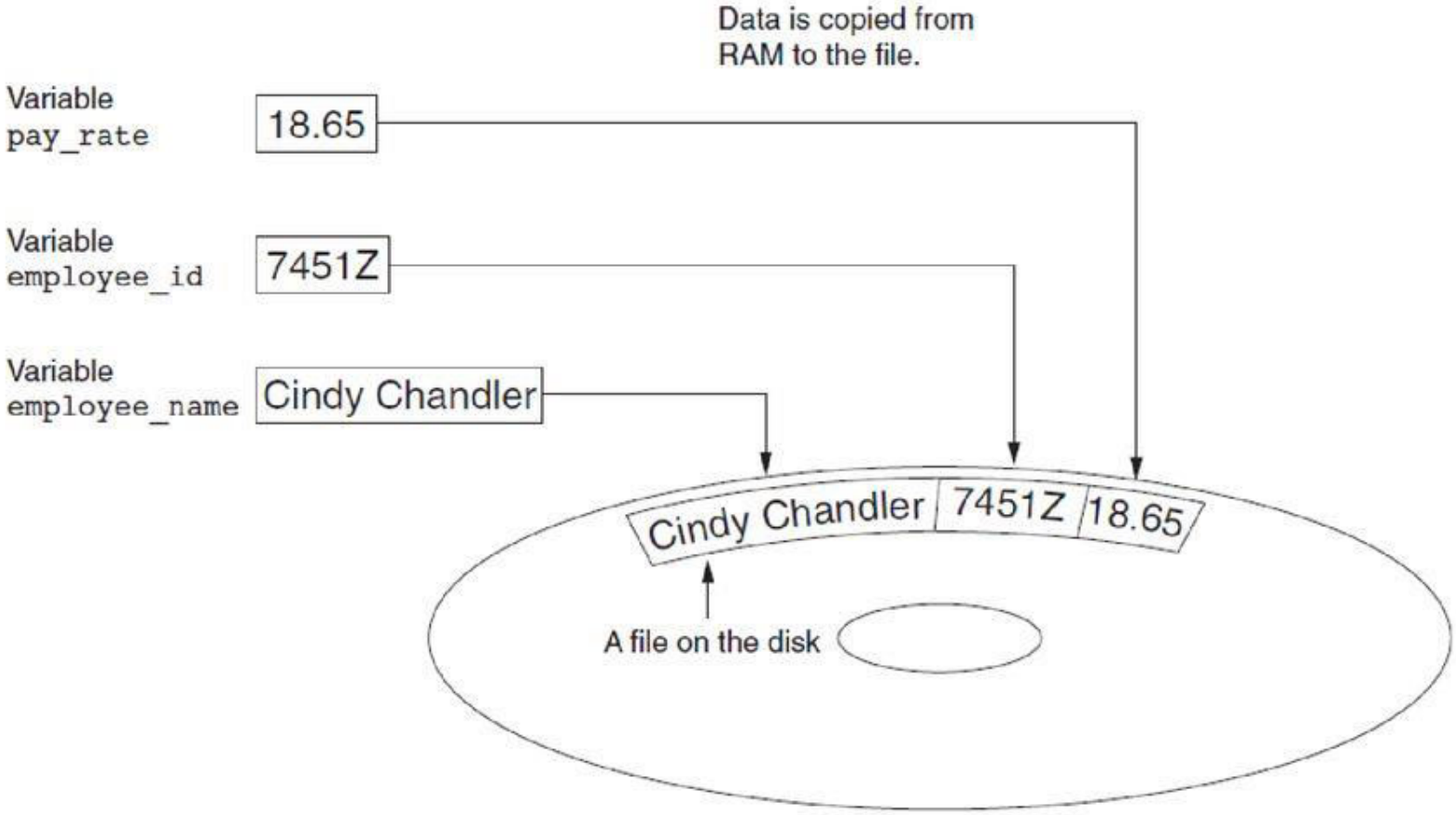
Topics

- Introduction to File Input and Output
- Using Loops to Process Files
- Processing Records
- Exceptions

Introduction to File Input and Output

- **For program to retain data between the times it is run, you must save the data**
  - Data is saved to a file, typically on computer disk
  - Saved data can be retrieved and used at a later time
- **"Writing data to": saving data on a file**
- **Output file: a file that data is written to**

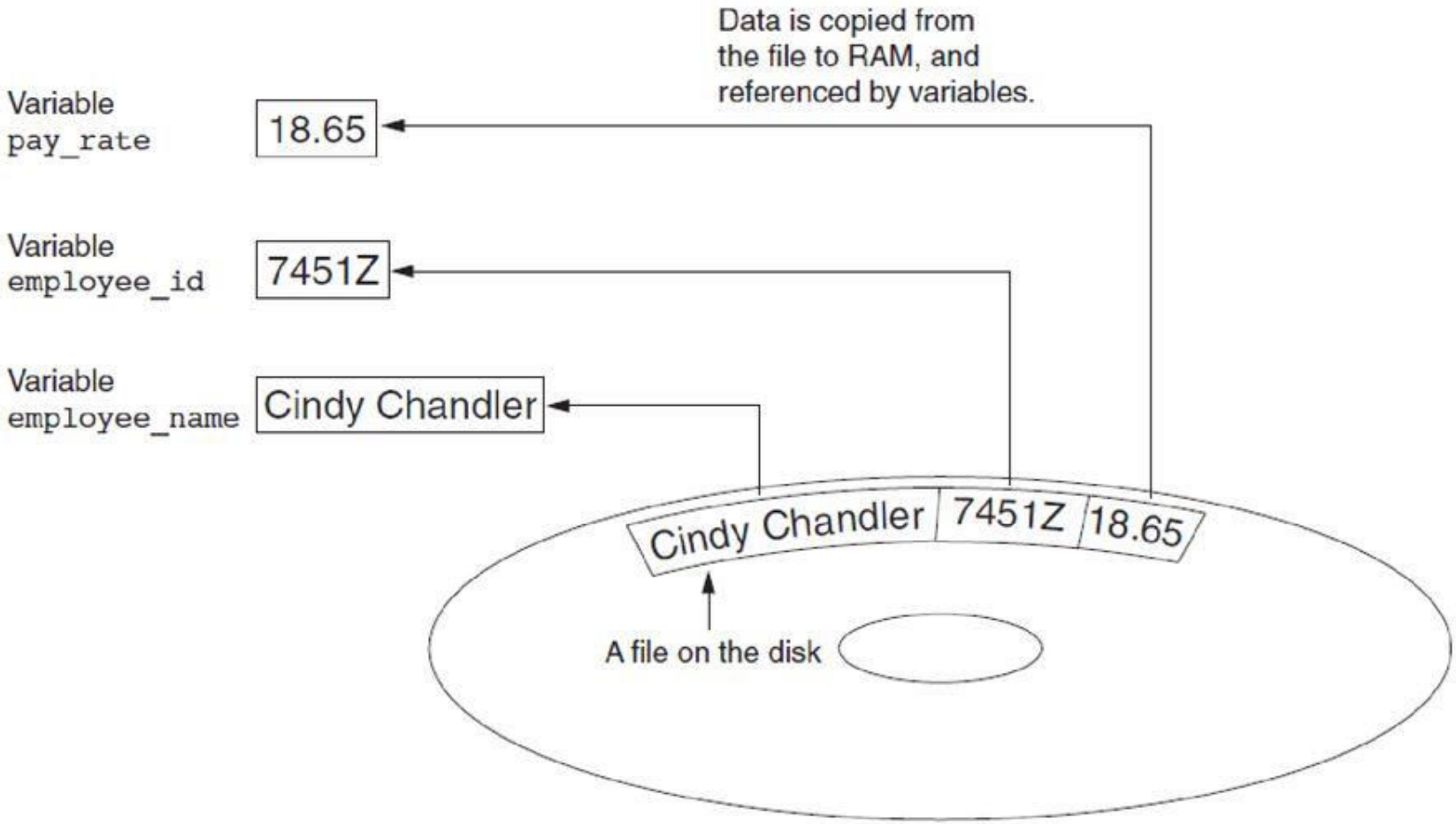
Figure 6-1 Writing data to a file



Introduction to File Input and Output (cont'd.)

- **"Reading data from": process of retrieving data from a file**
- **Input file: a file from which data is read**
- **Three steps when a program uses a file**
  - Open the file
  - Process the file
  - Close the file

Figure 6-2 Reading data from a file



Types of Files and File Access Methods

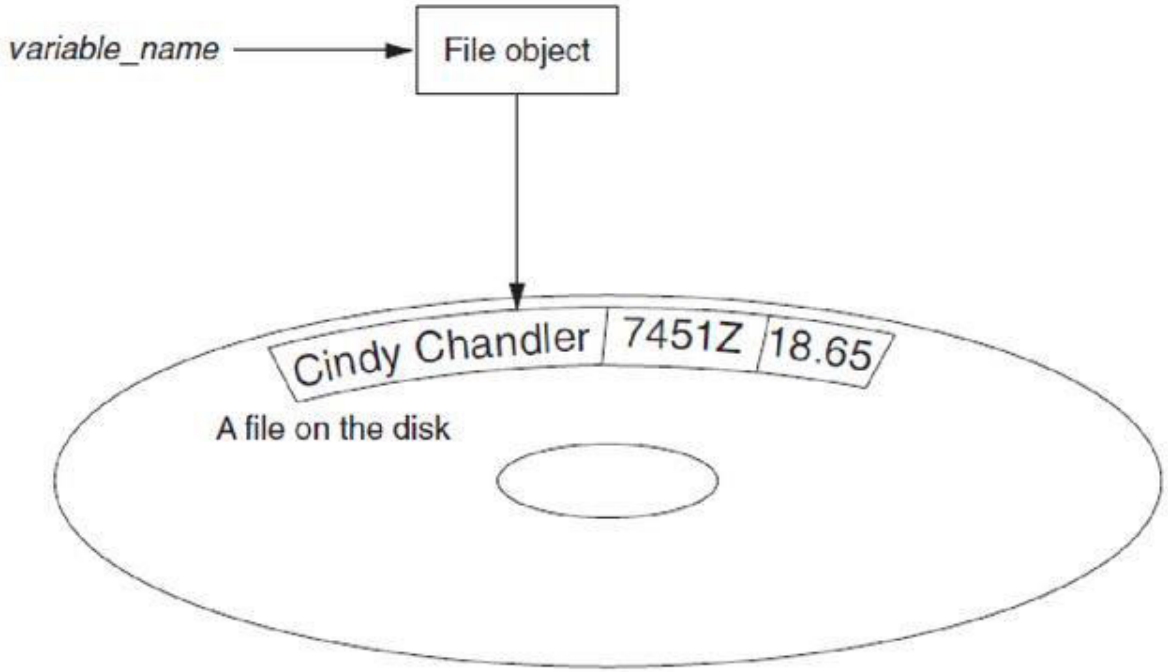
- **In general, two types of files**
  - Text file: contains data that has been encoded as text
  - Binary file: contains data that has not been converted to text
- **Two ways to access data stored in file**
  - Sequential access: file read sequentially from beginning to end, can't skip ahead
  - Direct access: can jump directly to any piece of data in the file

Filenames and File Objects

- **Filename extensions: short sequences of characters that appear at the end of a filename preceded by a period**
  - Extension indicates type of data stored in the file
- **File object: object associated with a specific file**
  - Provides a way for a program to work with the file: file object referenced by a variable

Filenames and File Objects (cont'd.)

Figure 6-4 A variable name references a file object that is associated with a file



Opening a File

- **`open` function: used to open a file**
  - Creates a file object and associates it with a file on the disk
  - General format:

    ```
    file_object = open(filename, mode)
    ```

- **Mode: string specifying how the file will be opened**
  - Example: reading only (`'r'`), writing (`'w'`), and appending (`'a'`)

Specifying the Location of a File

- If `open` function receives a filename that does not contain a path, assumes that file is in same directory as program
- If program is running and file is created, it is created in the same directory as the program
  - Can specify alternative path and file name in the `open` function argument
    - Prefix the path string literal with the letter `r`

Writing Data to a File

- **Method: a function that belongs to an object**
  - Performs operations using that object
- **File object's `write` method used to write data to the file**
  - Format: *`file_variable`*`.write(`*`string`*`)`
- **File should be closed using file object `close` method**
  - Format: *`file_variable`*`.close()`

Reading Data From a File

- **`read` method: file object method that reads entire file contents into memory**
  - Only works if file has been opened for reading
  - Contents returned as a string
- **`readline` method: file object method that reads a line from the file**
  - Line returned as a string, including `'\n'`
- **Read position: marks the location of the next item to be read from a file**

Concatenating a Newline to and Stripping it From a String

- **In most cases, data items written to a file are values referenced by variables**
  - Usually necessary to concatenate a `'\n'` to data before writing it
    - Carried out using the + operator in the argument of the `write` method
- **In many cases need to remove `'\n'` from string after it is read from a file**
  - `rstrip` method: string method that strips specific characters from end of the string

Appending Data to an Existing File

- **When open file with `'w'` mode, if the file already exists it is overwritten**
- **To append data to a file use the `'a'` mode**
  - If file exists, it is not erased, and if it does not exist it is created
  - Data is written to the file at the end of the current contents

Writing and Reading Numeric Data

- **Numbers must be converted to strings before they are written to a file**
- **`str` function: converts value to string**
- **Number are read from a text file as strings**
  - Must be converted to numeric type in order to perform mathematical operations
  - Use `int` and `float` functions to convert string to numeric value

Using Loops to Process Files

- **Files typically used to hold large amounts of data**
  - Loop typically involved in reading from and writing to a file
- **Often the number of items stored in file is unknown**
  - The `readline` method uses an empty string as a sentinel when end of file is reached
    - Can write a while loop with the condition

```
while line != ''
```

File Modes

Types of File Modes

File Mode	Description
r	Opens a file for reading only. The file pointer is placed at the beginning of the file. This is the default mode.
rb	Opens a file for reading only in binary format. The file pointer is placed at the beginning of the file. This is the default mode.
r+	Opens a file for both reading and writing. The file pointer placed at the beginning of the file.
rb+	Opens a file for both reading and writing in binary format. The file pointer placed at the beginning of the file.
w	Opens a file for writing only. Overwrites the file if the file exists. If the file does not exist, creates a new file for writing.
wb	Opens a file for writing only in binary format. Overwrites the file if the file exists. If the file does not exist, creates a new file for writing.
w+	Opens a file for both writing and reading. Overwrites the existing file if the file exists. If the file does not exist, creates a new file for reading and writing.
wb+	Opens a file for both writing and reading in binary format. Overwrites the existing file if the file exists. If the file does not exist, creates a new file for reading and writing.

a	Opens a file for appending. The file pointer is at the end of the file if the file exists. That is, the file is in the append mode. If the file does not exist, it creates a new file for writing.
ab	Opens a file for appending in binary format. The file pointer is at the end of the file if the file exists. That is, the file is in the append mode. If the file does not exist, it creates a new file for writing.
a+	Opens a file for both appending and reading. The file pointer is at the end of the file if the file exists. The file opens in the append mode. If the file does not exist, it creates a new file for reading and writing.
ab+	Opens a file for both appending and reading in binary format. The file pointer is at the end of the file if the file exists. The file opens in the append mode. If the file does not exist, it creates a new file for reading and writing.

File Methods

File Methods	Description	Prototype
close()	Closes the file	fileobject.close()
flush()	Flushes the internal buffer	fileobject.flush()
read()	Returns the file content	data = fileobject.read()
readable()	Returns whether the file stream can be read or not	boolean = fileobject.flush()
readline()	Returns one line from the file	data = fileobject.readline()
readlines()	Returns a list of lines from the file	filelines = fileobject.read()
seek()	Change the file position	filelines = fileobject.seek(offset)
seekable()	Returns whether the file allows us to change the file position	bool = fileobject.seekable()
tell()	Returns the current file position	position = fileobject.tell()
truncate()	Resizes the file to a specified size	fileobject.truncate(size)
writeable()	Returns whether the file can be written to or not	bool = fileobject.writeable()
write()	Writes the specified string to the file	fileobject.write(data)
writelines()	Writes a list of strings to the file	fileobject.writelines(listofstrings[])

Summary

- A file is a container in a computer system for storing information.
- **Binary file :** Binary file is a collection of bytes or a character stream.
- **Text file :** A text file is a stream of characters that can be processed sequentially and logically in the forward direction.
- Creation of a new file
- Modification of data or file attributes
- Reading of data from the file
- Opening the file in order to make the contents available to other programs
- Writing data to the file
- Closing or terminating a file operation


Thank
you