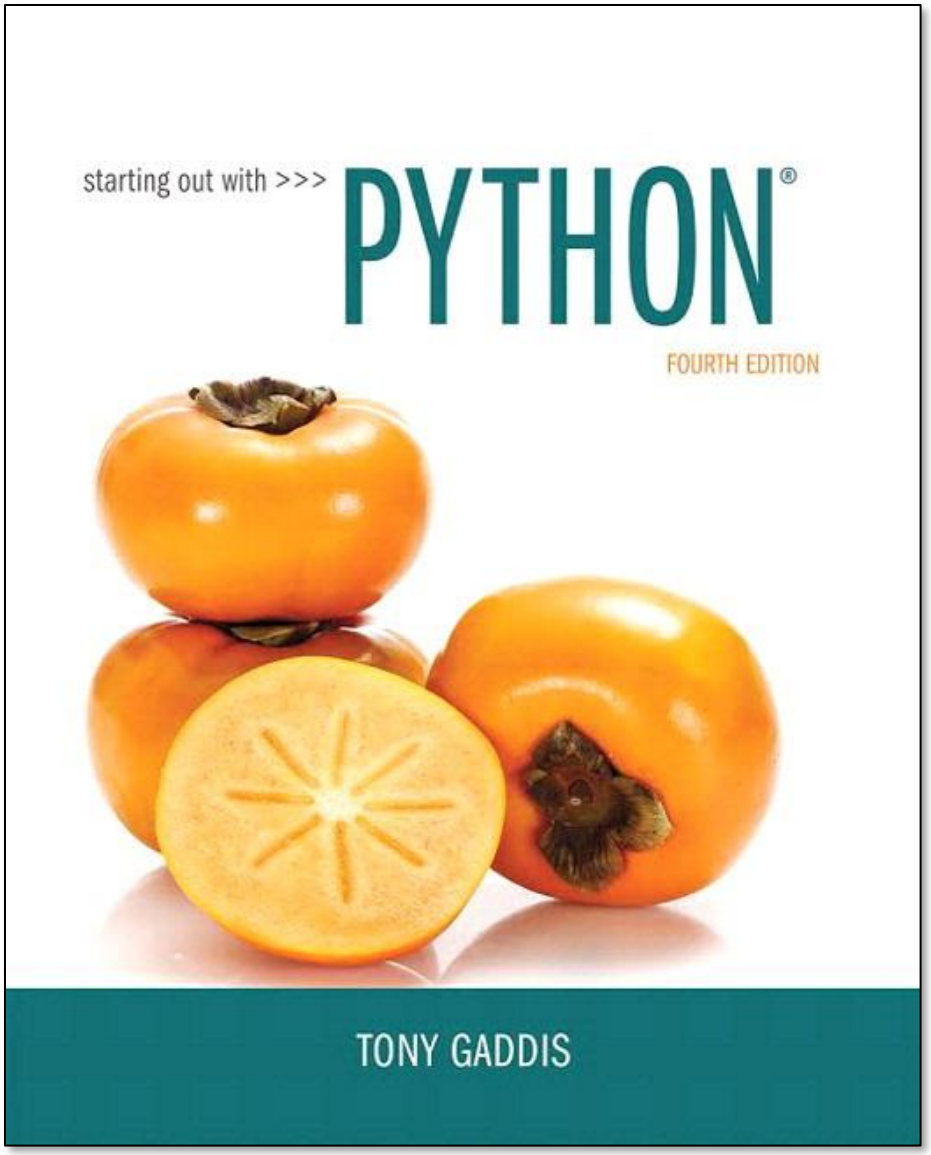
CHAPTER 6

Files and Exceptions



Dr. Bita Shadgar

Topics

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

Learning Outcomes

- At the end of this week the students must be able to:
  - Describe how data can be stored/retrieved to/from a file
  - Define different types of files and different types of access to a file
  - Open existing file and use it as input resource
  - Use loops to process files
  - Create a file and use it as an output resource
  - Use a file as input and output resource
  - Append data to the end of an existing file

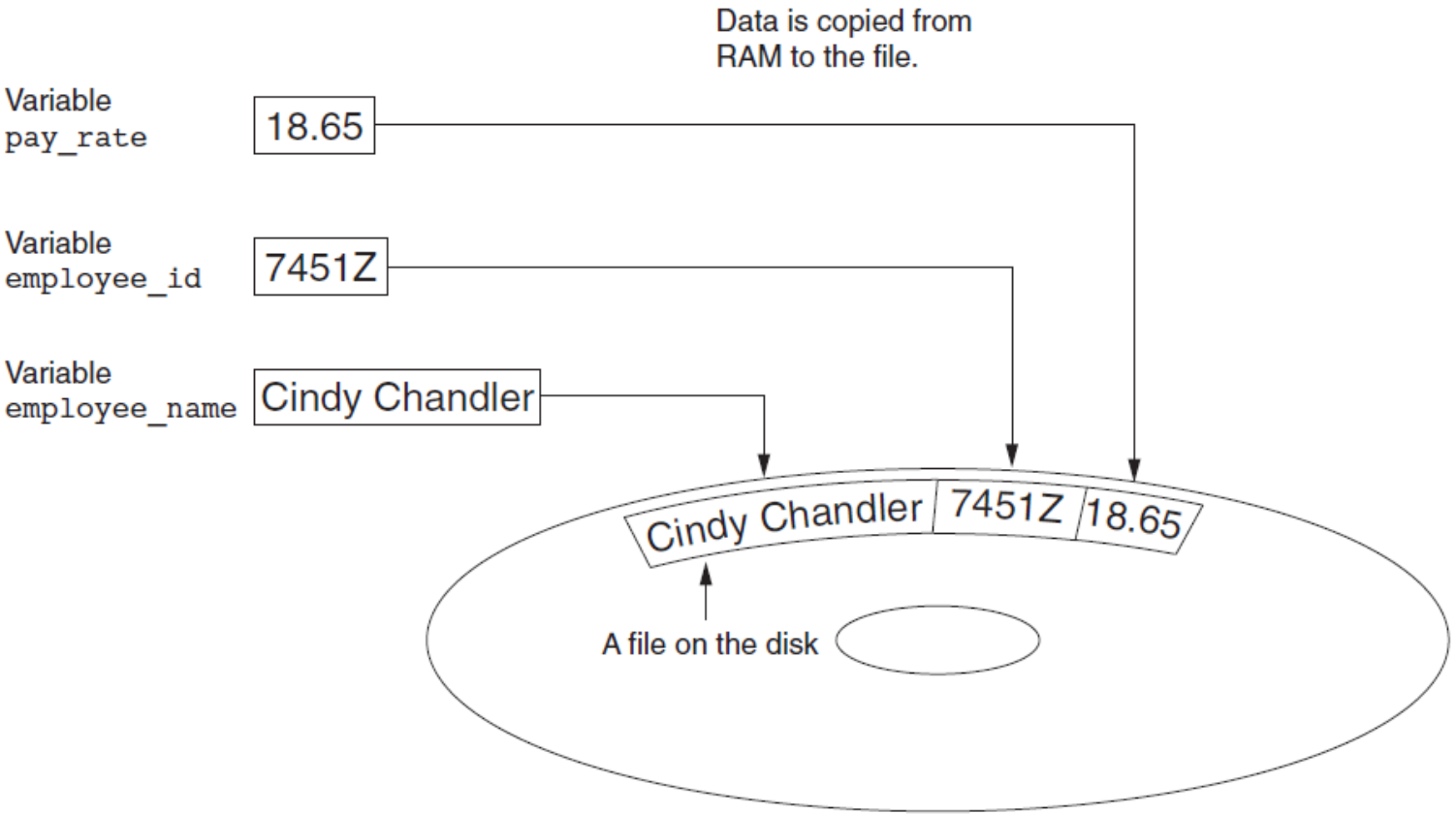
Introduction to File and Input/Output

- For program to retain data between the times it is run, you must save and retrieve the data
  - Data is saved to a file, typically on computer disk
  - Saved data can be retrieved and used later
- Three steps when a program uses a file
  - Open the file
  - Process the file
  - Close the file

File and Output

- **"Writing data to":** saving data on a file
- **Output file**: a file that data is written to

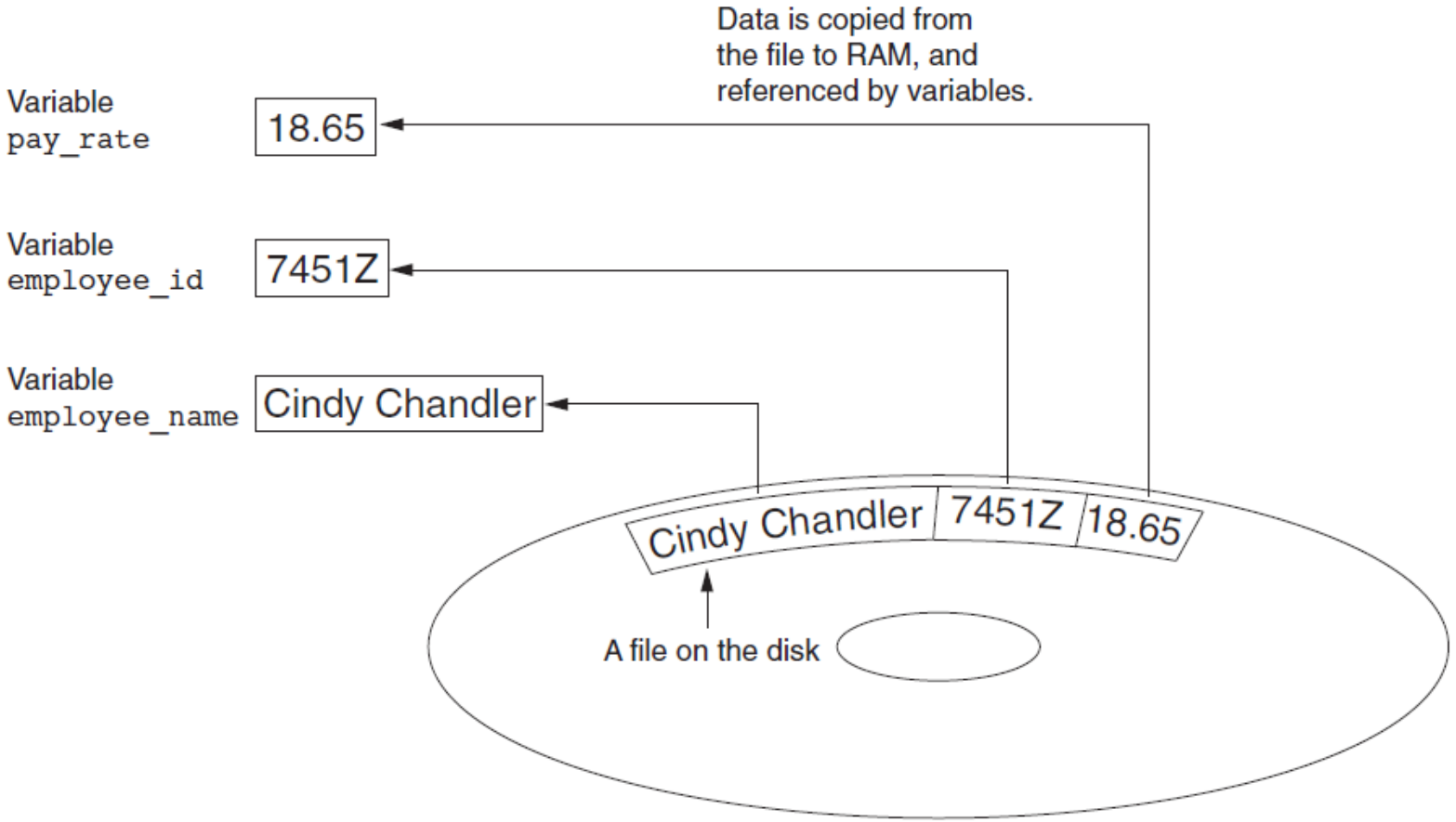
Figure 6-1 Writing data to a file



File and Input

- **"Reading data from":** retrieving data from a file
- **Input file**: a file from which data is read

Figure 6-2 Reading data from a file



Types of Files and File Access Methods

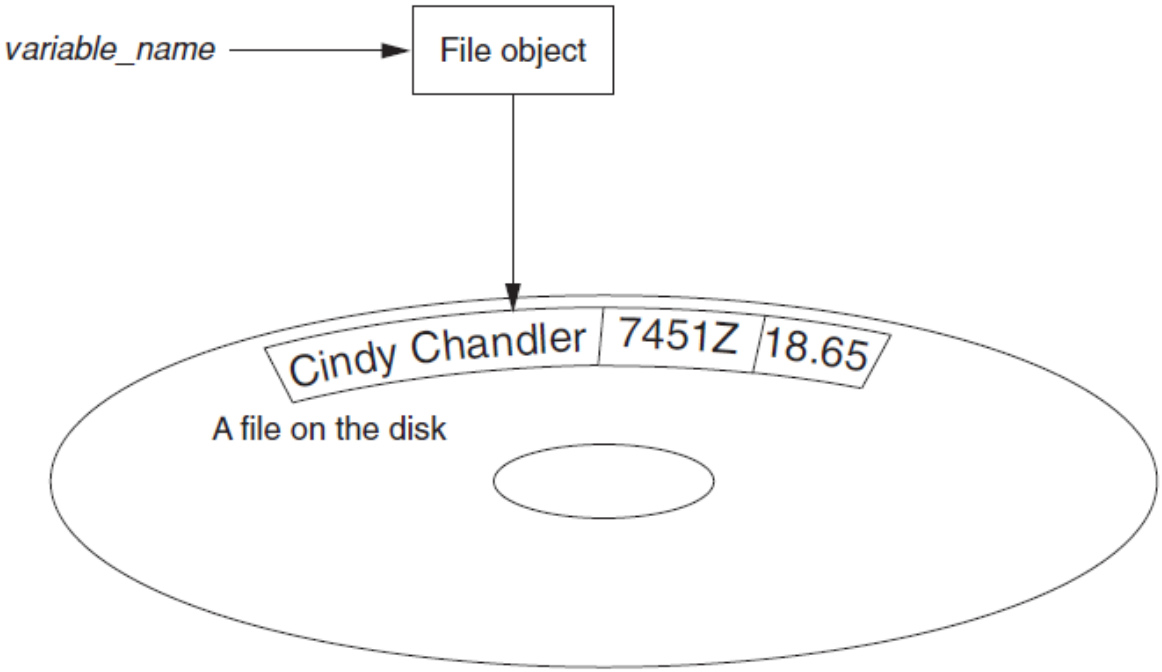
- In general, two types of files
  - **Text file**: contains data that has been encoded as text
  - **Binary file**: contains data that has not been converted to text
- Two ways to access data stored in file
  - **Sequential access**: file read sequentially from beginning to end, can't skip ahead
  -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

Filenames and File Objects

- **Filename** is normally composed of a name and an extension
  - Filename is used for recognizing file on disk
- **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
  - This is used for recognizing file in program

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'`)

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()`
    - Hands-on : open file_write.py and run it.
    - open write_numbers.py and run it.

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 a 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
  - Use absolute path to have different path

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
    - Practice with file_read.py
    - Modify the file to get the name of file from user (save as display_file.py)

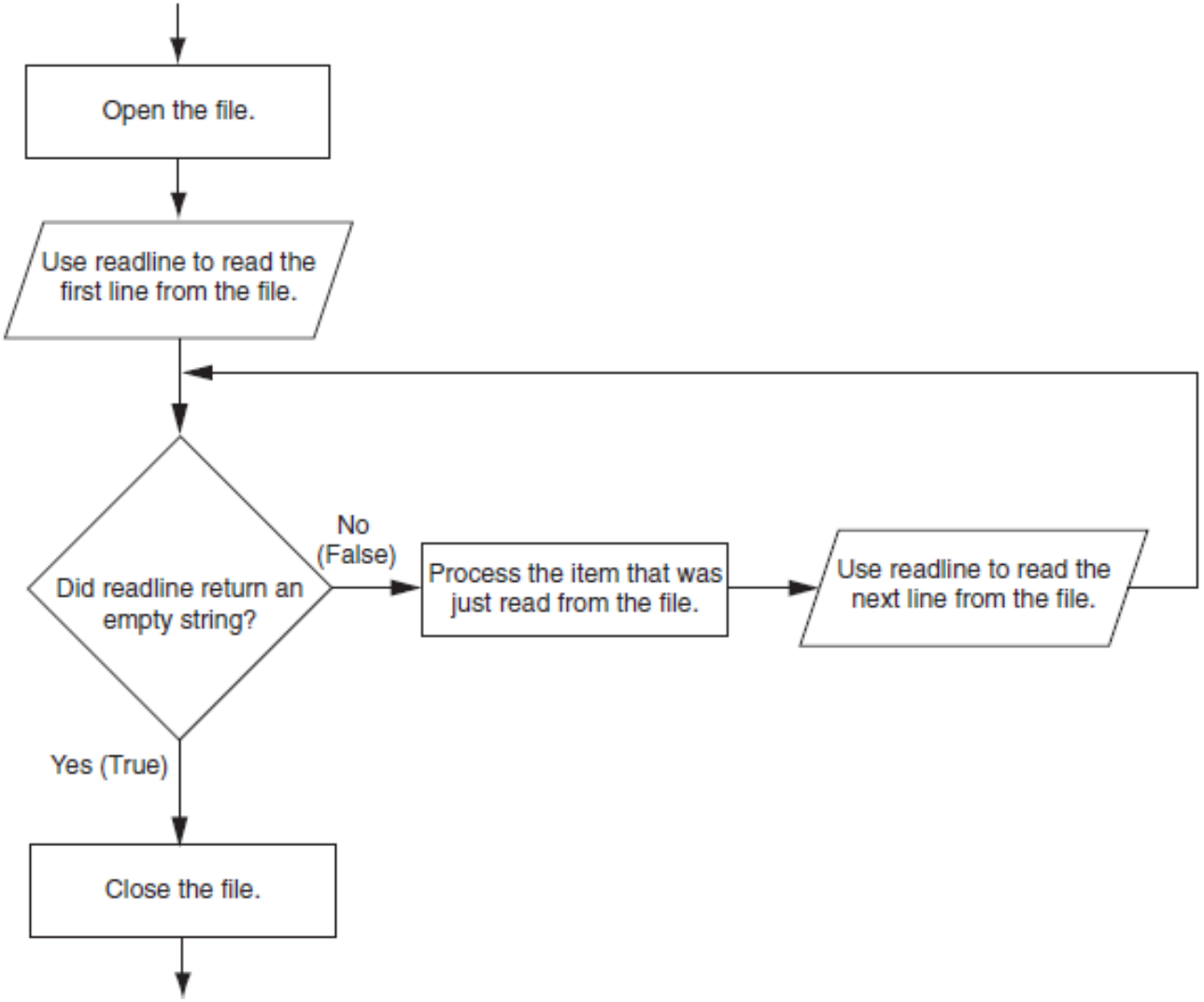
Reading Data From a File

- **`readline` method**: file object method that reads a line from the file
  - Line returned as a string, including `'\n'`
    - Practice with Line_read.py
  - readline returns a string value and it must be converted to the right type before using it
    - Practice with Line_numbers.py

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 != ''`
    - Practice with read_sales.py

Figure 6-17 General logic for detecting the end of a file



Processing Records

- **Record:** set of data that describes one item
- **Field:** single piece of data within a record
- Write record to sequential access file by writing the fields one after the other
- Read record from sequential access file by reading each field until record complete

Concatenating/Stripping a Newline to/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
    - Practice with read_emp_records.py

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 Python's `for` Loop to Read Lines

- Python allows the programmer to write a `for` loop that automatically reads lines in a file and stops when end of file is reached
  - Format: `for` *`line`* `in` *`file_object`*`:`
    `statements`
  - The loop iterates once over each line in the file
    - Practice with read_running_times.py

Summary

- Types of files and file access methods
- Filenames and file objects
- Writing data to a file
- Reading data from a file and determining when the end of the file is reached
- Processing records

More Practice

- Check out review questions in chapter 6 of the textbook including :
  - Multiple Choices,
  - True or False
  - Short Answer
  - Algorithm WorkBench
  - Programming Exercises (1-8)