# Files

# Files

- Files are persistent data storage so that data is available for a program to read next time.
- You use the open(file, mode) to open a file. The open() returns a file object for the specified file with the specified node.
  - "r" - for reading an existing file.
  - "w" - for creating a new file or erasing all the data in an existing file.
  - "a" - for appending the data to the end of the existing file.
- You use the close() to close the file to free all resources.
- When you use a with statement to open a file, it automatically closes the file after executing its block of statements. That dress all resources used by the file, even if an exception occurs and the program ends prematurely.

```
def writeCourses(self, courses):
        with open(self.__filename, "w") as file:
            for course in courses:
                file.write(course + "\n")
```

```
fo = open("data.txt", "r")
line = fo.readline()
print "Read Line: %s" % (line)
line = fo.readline(5)
print "Read Line: %s" % (line)
fo.close()
```

# Writing and Reading a Text File

- You can use the write(str) method to write data to a text file. If you want to start a new line, you must include the new line character.
- You can use the following methods to read data from a file:
    - read() - to read the entire file and returns its contents as a string.
    - readlines() - to read the entire file and returns it a list.
    - readline() - to read the next line in the file and returns its content as a string.

```
def readCourses(self):
        courses = []
        with open(self.__filename) as file:
            for line in file:
                line = line.replace("\n", "")
                courses.append(line)
        return courses
```

**This example uses a for loop to read each line of a file**

```
fo = open("data.txt", "r")
line = fo.readline()
print "Read Line: %s" % (line)
line = fo.readline(5)
print "Read Line: %s" % (line)
fo.close()
```

**This example uses readline() to read each line one by one**

## Writing and Reading a List

- When you read a text file into a list, you typically want to remove the new line character that's at the end of each line. to do that, you can use the replace() method of a string object.
- Before you can write a non-string value to a text file, you must convert it to a string value. Later, you read that string value, you can convert it back to its original data type.
- For more about files, please refer to the following link:
  - Input and Output

```
def readCourses(self):
        courses = []
        with open(self.__filename) as file:
            for line in file:
                line = line.replace("\n", "")
                courses.append(line)
        return courses
```

# CSV Files

- A CSV (common-separated values) file stores multiple values per line, typically using commas to separate each value.
- we can treat each line as a row, and each row contains one or more columns.
- You can use the writer() method of the csv module to get a writer object. Then, you can then use writerows(rows) to write data.
- When you open a CSV file for reading or writing, you typically specify an argument named new line with a value of an empty string. This enables universal new lines mode, so the reading and writing operations work correctly for all operating systems.
- You can use the reader() method to create a reader object.Then, you can use a for statement with the reader object to read the data in the file.
- By default, reader and writer objects use commas to delimit the columns of a row and only add quotes to columns when necessary. But, when you create reader and writer object, you can specify arguments that change the delimiter.
- For details, please refer to this link:
  - CSV File Reading and Writing

# CSV File - Examples

**courses.csv**

```
CS480,Java Programming
CS526,Advanced Web Programming
CS557,Advanced JavaScript Programming
```

```
def writeCourses(self, courses):
        with open(self.__filename, "w", newline="") as file:
            writer = csv.writer(file)
            writer.writerows(courses)

    def readCourses(self):
        courses = []
        with open(self.__filename, newline="") as file:
            reader = csv.reader(file)
            for row in reader:
                courses.append(row)
        return courses
```

# Example: test_text.py

## courses.txt

```
CS480 Java Programming
CS526 Advanced Web Programming
CS557 Advanced JavaScript Programming
```

```
class CourseFile:
    def __init__(self, filename):
        self.__filename = filename;

    def writeCourses(self, courses):
        with open(self.__filename, "w") as file:
            for course in courses:
                file.write(course + "\n")

    def readCourses(self):
        courses = []
        with open(self.__filename) as file:
            for line in file:
                line = line.replace("\n", "")
                courses.append(line)
        return courses

    def listCourses(self, courses):
        for i in range(len(courses)):
            print(i+1, courses[i])
        print()

    def addCourse(self, courses):
        course = input("Course: ")
        courses.append(course)
        self.writeCourses(courses)
        print(course + " was added.\n")

    def deleteCourse(self, courses):
        index = int(input("Item no: "))
        if index < 1 or index > len(courses):
            print('Invalid course no!')
            return
        course = courses.pop(index - 1)
        self.writeCourses(courses)
        print(course + " was deleted.\n")

def displayMenu():
    print("The Course List program")
    print()
    print("COMMAND MENU")
    print("L - List all courses")
    print("A - Add a course")
    print("D - Delete a course")
    print("E - Exit program")
    print()

def main():
    file = CourseFile("courses.txt")
    displayMenu()
    courses = file.readCourses()
    while True:
        command = input("Command: ")
        command = command.lower()
        if command == "l":
            file.listCourses(courses)
        elif command == "a":
            file.addCourse(courses)
        elif command == "d":
            file.deleteCourse(courses)
        elif command == "e":
            print("Bye!")
            break
        else:
            print("Not a valid command. Please try again.")

if __name__ == "__main__":
    main()
```

# Example: test_csv.py

**courses.csv**

```
CS480,Java Programming
CS526,Advanced Web Programming
CS557,Advanced JavaScript Programming
```

```
import csv

class CourseFile:
    def __init__(self, filename):
        self.__filename = filename;

    def writeCourses(self, courses):
        with open(self.__filename, "w", newline="") as file:
            writer = csv.writer(file)
            writer.writerows(courses)

    def readCourses(self):
        courses = []
        with open(self.__filename, newline="") as file:
            reader = csv.reader(file)
            for row in reader:
                courses.append(row)
        return courses

    def listCourses(self, courses):
        for i in range(len(courses)):
            print(i+1, courses[i])
        print()

    def addCourse(self, courses):
        courseNo = input("Course No: ")
        courseTile = input("Course Title: ")
        course = []
        course.append(courseNo)
        course.append(courseTile)
        courses.append(course)
        self.writeCourses(courses)
        print(courseNo + " was added.\n")

    def deleteCourse(self, courses):
        index = int(input("Item no: "))
        if index < 1 or index > len(courses):
            print('Invalid course no!')
            return
        course = courses.pop(index - 1)
        self.writeCourses(courses)
        print(course[0] + " was deleted.\n")

def displayMenu():
    print("The Course List program")
    print()
    print("COMMAND MENU")
    print("L - List all courses")
    print("A - Add a course")
    print("D - Delete a course")
    print("E - Exit program")
    print()

def main():
    file = CourseFile("courses.csv")
    displayMenu()
    courses = file.readCourses()
    while True:
        command = input("Command: ")
        command = command.lower()
        if command == "l":
            file.listCourses(courses)
        elif command == "a":
            file.addCourse(courses)
        elif command == "d":
            file.deleteCourse(courses)
        elif command == "e":
            print("Bye!")
            break
        else:
            print("Not a valid command. Please try again.")

if __name__ == "__main__":
    main()
```