

Introduction to Computer and Programming

Lecture 6: Advanced Topics, Files, Project





Table of Contents

- String Functions
- Sorting and Filtering
- Files
- Project



String Functions



Basic String Operations

```
message = 'Hello to Python World'
```

```
print(message, len(message))
```

```
print(message[0], message[len(message)-1], message[-5])
```

```
print("%s - %s - %s"%(message[9:], message[:15], message[9:15]))
```

```
message = "Hello"*3; print(message)
```

```
message = "{} * {} = {}".format(7, 5, 7 * 5); print(message)
```

```
message = "%d / %d = %f"%(7, 5, 7 / 5); print(message)
```

```
message = "Hello 'Mohamed'"; print(message)
```

```
message = 'Hello "Mohamed"'; print(message)
```

```
message = "Hello \"Mohamed\""; print(message)
```



String Functions

```
message = 'hello to Computer world. Computer is amazing.'
print(message)
message = message.title(); print(message)
print(message.count('Computer'))
print(message.replace('Computer', 'Python'))
print(message.lower()); print(message.upper())
print(message.capitalize());
print(message.find('Computer'), message.index('Computer'))
print(message.startswith('Hello'), message.endswith('.'))
print(message.split(' '))
print('##Egypt##'.strip('#'))
print('##Egypt##'.lstrip('#'), '##Egypt##'.rstrip('#'))
```



String Example

```
text = input("Enter values to sum:")
textValues = text.split(" ")
sum = 0
for textValue in textValues:
    sum += int(textValue)
print("Sum:", sum)
```



Sorting and Filtering



Sorting

```
numbers = [20, 130, 40, 70, 12, 77, 15]
numbers = sorted(numbers)
print(numbers)
```

```
names = ['Ahmed', 'Walaa', 'Safaa', 'Magdy']
names = sorted(names, reverse = True);
print(names)
```

```
names = ['Ahmed', 'walaa', 'sAfaa', 'Magdy']
names = sorted(names, key = lambda k: k.lower());
print(names)
```



Dictionary Sort

```
degrees = {'math': 80, 'physics': 100, 'history': 70}
items = degrees.items(); print(items)
items = sorted(items, key = lambda item: item[1]); print(items)
degrees = dict(items); print(degrees)
```

```
employees = [{'name': 'Ahmed', 'salary':1200},
              {'name': 'Saleh', 'salary':1000},
              {'name': 'soaad', 'salary':1800}]
employees = sorted(employees, key = lambda item: item['salary']);
print(employees)
employees = sorted(employees, key = lambda item: item['name'].lower());
print(employees)
```



Filter

```
numbers = [20, 130, 40, 70, 12, 77, 15]
numbers = filter(lambda value: value>50, numbers)
numbers = list(numbers); print(numbers)
```

```
names = ['Ahmed', 'Saleh', 'Ali', 'amer', 'Salem']
names = filter(lambda value: value.lower().startswith('a'), names)
names = list(names); print(names)
```



Exercise

- Try support all following separators: comma, semicolon, column, space and tab
- What if you have multiple spaces? Propose a solution?
- Change the program to accept float
- Write a program that collect user information as following:
 - Please provide me with your name, age and weight?
 - My name is Ahmed Ali, 17 years old, 66.5 KG
 - My name is Safaa Said, 55 KG, 18 years old
- Try sort those word by their length (the, in, yesterday, africa)



Files



Files

```
file = open(".\\names.txt", "r")
lines = file.readlines()
for line in lines:
    print(line.strip())
file.close()
```

Read

```
file = open(".\\names.txt", "w")
while True:
    name = input("Enter Name:")
    if len(name)>0:
        file.write("%s\n"%(name))
    else: break
file.close()
```

Write

Histogram Example 1/4

```
file = open(".\\book.txt", "r")
words = file.read().split()
for word in words:
    letters = filter(lambda i: i.isalpha(), word.lower())
    word = ''.join(list(letters))
    print(word)
```



Histogram Example 2/4

```
file = open(".\\book.txt", "r")
words = file.read().split()
histogram = {}
for word in words:
    letters = filter(lambda i: i.isalpha(), word.lower())
    word = ''.join(list(letters))
    print(word)
    if len(word)==0: continue
    if word in histogram: histogram[word] += 1
    else: histogram[word] = 1

print(histogram)
```



Histogram Example 3/4

```
histogram = dict(sorted(histogram.items(), key=lambda x:x[1], reverse=True))
```

```
histogram = dict(list(filter(lambda x: x[1]>100, histogram.items())))
```

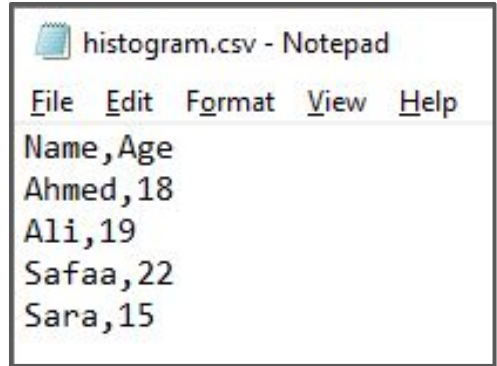
```
for word in histogram:
```

```
    print("{}\t:{}".format(word, histogram[word]))
```



CSV Files

- Type of organized text files
- Data is arranged in tabular form
- Data are comma separated
- First row is considered as a header
- Used to transfer data between computers and users
- Used to prepare data for analysis



```
histogram.csv - Notepad
File Edit Format View Help
Name, Age
Ahmed, 18
Ali, 19
Safaa, 22
Sara, 15
```

	A	B
1	Name	Age
2	Ahmed	18
3	Ali	19
4	Safaa	22
5	Sara	15
6		

Histogram Example 4/4

```
opfile = open(".\\histogram.csv", "w")
opfile.write("Word, Frequency\n")
for word in histogram:
    opfile.write("{}\n".format(word, histogram[word]))
opfile.close()
```



JSON



JSON Format

- Type of organized text files
- Represented as key:value pairs
- Allow nested structures and arrays
- Used to transfer data between computers
- Used to prepare data for analysis

```
1  [{
2      "name": "Ahmed Ali",
3      "age": 15,
4      "salary": 18000,
5      "address": {
6          "country": "Egypt",
7          "city": "Cairo"
8      }
9  }, {
10     "name": "Saraa Ahmed",
11     "age": 18,
12     "salary": 12000,
13     "address": {
14         "country": "Egypt",
15         "city": "Alexandria"
16     }
17 }]
```



Reading JSON Files

```
import json

file = open(".\\users.json", "r")
content = file.read()
users = json.loads(content)
for user in users:
    print("Name:{}".format(user['name']))
    print("Age:{}".format(user['age']))
    print("Address:{}, {}".format(user['address']['city'], user['address']['country']))
    print()
file.close()
```



Writing to JSON Files

...

```
opfile = open(".\\final_users.json", "w")
users += [{'name': 'Shady', 'age': 19, 'salary': 15000, 'address':
{'city': 'Tanta', 'country': 'Egypt'}}}]
content = json.dumps(users)
opfile.write(content)
opfile.close()
```



HTML



HTML Format

- Used to format content for web.
- Allow adding color, fonts, table, images, list and more.
- Consists of set of tags each has special effect



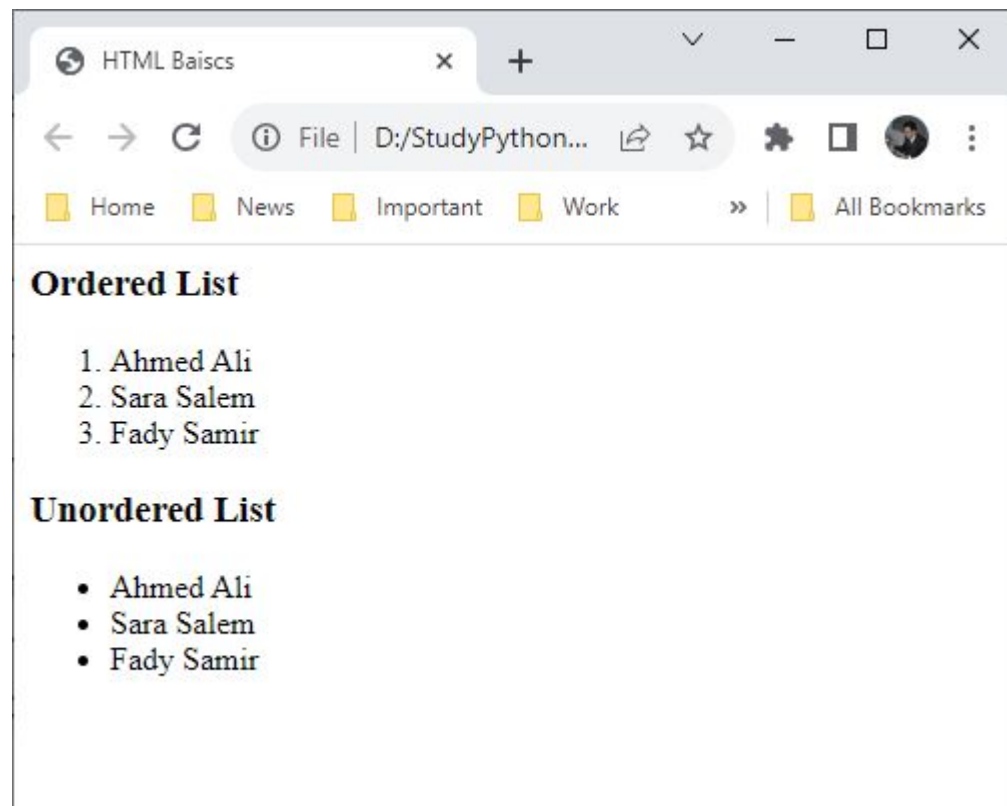
Simple HTML Document

```
<html>
  <head>
    <title>HTML Basics</title>
  </head>
  <body>
    <p>Hello <b>every</b> <i>one</i>.</p>
    <p>This is an <font color="red">HTML</font> sample.</p>
    <h1>This is a H1 Title</h1>
    <h2>This is a H2 Title</h2>
    <h3>This is a H3 Title</h3>
    <h4>This is a H4 Title</h4>
  </body>
</html>
```



Showing a Lists

```
<body>
  <h3>Ordered List</h3>
  <ol>
    <li>Ahmed Ali</li>
    <li>Sara Salem</li>
    <li>Fady Samir</li>
  </ol>
  <h3>Unordered List</h3>
  <ul>
    <li>Ahmed Ali</li>
    <li>Sara Salem</li>
    <li>Fady Samir</li>
  </ul>
</body>
```



Showing Tables

```
<body>
  <h3>Employees Table</h3>
  <table border="1px" width="300px">
    <tr><th>Name</th><th>Age</th><th>Salary</th></tr>
    <tr><td>Ahmed</td><td>35</td><td>30000</td></tr>
    <tr><td>Sara</td><td>25</td><td>15000</td></tr>
    <tr><td>Shady</td><td>30</td><td>25000</td></tr>
  </table>
</body>
```



Generate Users List Page

```
import json

file = open(".\\users.json", "r"); content = file.read(); users = json.loads(content)

body = "<h3>Users</h3>"
items = ""
for user in users:
    items += "<li>{}</li>".format(user['name'])
body += "<ol>{}</ol>".format(items)
html = "<html><body>{}</body></html>".format(body)

opfile = open(".\\users.html", "w"); opfile.write(html)
```



Generate Users Table Page (1/3)

```
import json

file = open(".\\users.json", "r")
content = file.read()
users = json.loads(content)
file.close()
```



Generate Users Table Page (2/3)

```
body = "<h3>Users Table</h3>"
```

```
rows = "<tr><th>Name</th><th>Age</th><th>Salary</th></tr>\n"
```

```
for user in users:
```

```
    rows += "<tr><td>{}</td><td>{}</td><td>{}</td></tr>\n"  
    .format(user['name'], user['age'], user['salary'])
```

```
body += "<table border='1px' width='50%'>{}</table>\n".format(rows)
```



Generate Users Table Page (3/3)

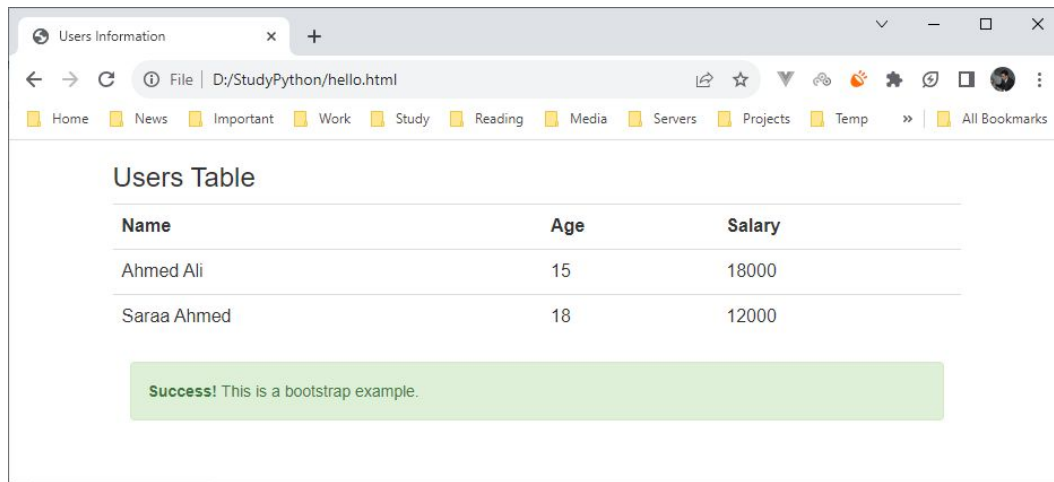
```
head = "<title>Users Information</title>"
html = "<html><head>{}</head><body>{}</body></html>".format(head, body)

opfile = open(".\\users.html", "w");
opfile.write(html)
opfile.close()
```



Enhance HTML More with Styles and Bootstrap

```
<html>
  <head>
    <title>Users Information</title>
    <link rel="stylesheet"
      href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
  </head>
  <body>
    <div class="container">
      <div class="row">
        <h3>Users Table</h3>
        <table class='table'>
          <tr><th>Name</th><th>Age</th><th>Salary</th></tr>
          <tr><td>Ahmed Ali</td><td>15</td><td>18000</td></tr>
          <tr><td>Saraa Ahmed</td><td>18</td><td>12000</td></tr>
        </table>
      </div>
      <div class="alert alert-success">
        <strong>Success!</strong> This is a bootstrap example.
      </div>
    </div>
  </body>
</html>
```



Project



Project Description

- Create Students Program
- Can add/edit/remove students information (code, name, birthdate)
- Students information are saved to json file “students.json”
- Can add/edit/remove courses information (code, name, max degree)
- Courses information are saved to json file “courses.json”
- Can supply grades to students per course.
- Grades information are saved to csv file “<course-code>.csv”
- Prints students result to html file “<student-code>.html”
- Course results are calculated using credit hours grading system.
- Generate bar chart for students results per course.
- Generate pie chart for course registration.



Project Rules

- 20 Degree
- Individual Project
- Discussion During Practical Examination Week
- Student will demonstrate the project.
- Student will explain code.
- Student will make a modification during the practical examination.



Working with Matplot library - Bar Charts

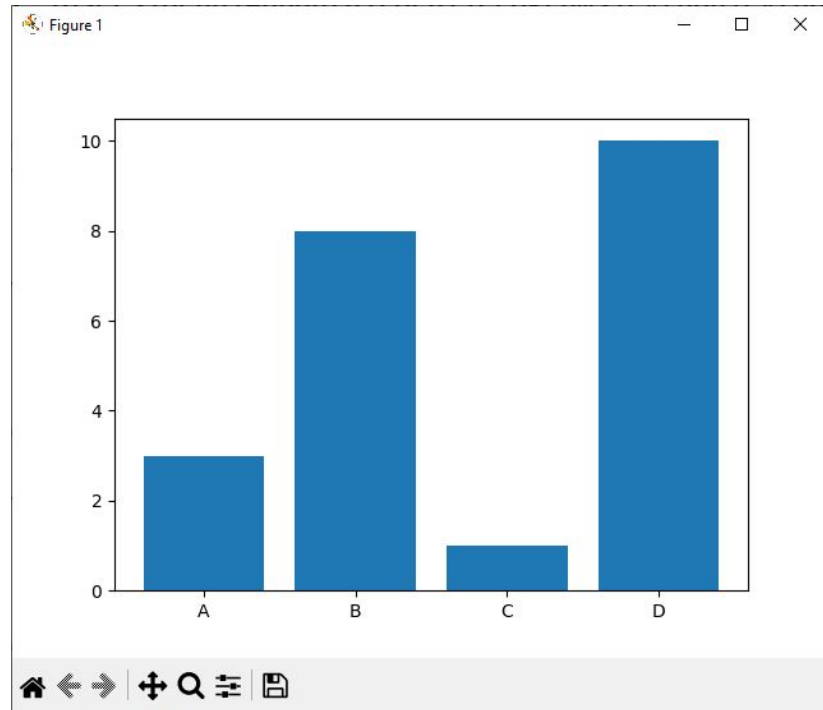
```
import matplotlib.pyplot as plt
```

```
x = ["A", "B", "C", "D"]
```

```
y = [3, 8, 1, 10]
```

```
plt.bar(x, y)
```

```
plt.show()
```



Working with Matplotlib library - Pie Charts

```
import matplotlib.pyplot as plt
```

```
x = ["A", "B", "C", "D"]
```

```
y = [3, 8, 1, 10]
```

```
plt.pie(y, labels=x)
```

```
plt.show()
```

