Programming for Business Computing

Making Statistical Plots with `matplotlib`

Ling-Chieh Kung

Department of Information Management
National Taiwan University

Making plots with `matplotlib`

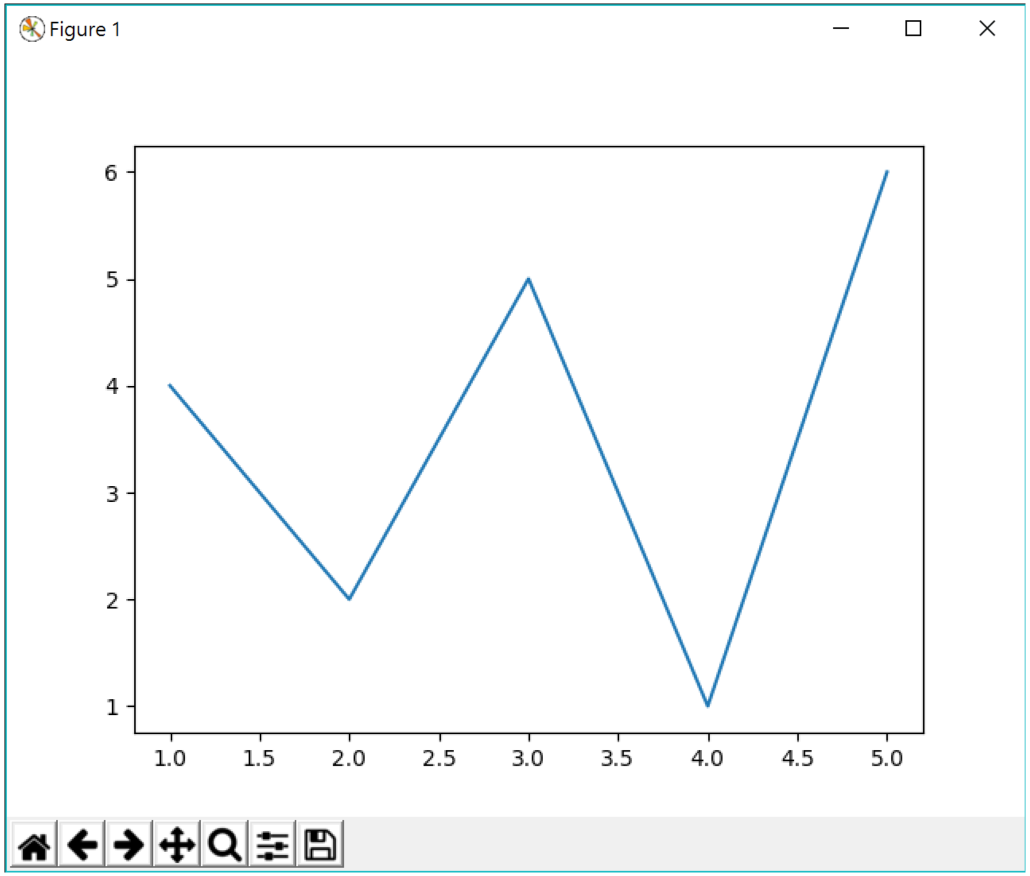
- In many cases, we want to make **plots**.
  - **Information visualization**.
- We will introduce how to use **`matplotlib`**, an open-source Python library, to make basic statistical plots.
  - Histograms, line charts, bar charts, pie charts, and scatter plots.
  - http://matplotlib.org/index.html
- First, we need to install **`matplotlib`**.
  - http://matplotlib.org/users/installing.html
  - Open your console/terminal/cmd, and execute:

```
python -m pip install -U pip setuptools
python -m pip install matplotlib
```

  - Do not forget to set the PATH variable or go to your Python script directory.

Testing matplotlib

```
import matplotlib.pyplot as py

x = range(1, 6)
y = [4, 2, 5, 1, 6]
py.plot(x, y)
py.show()
```



Data source

- To introduce the plotting tools in **`matplotlib`**, let's consider the data set contained in "midterm2.csv".

```
SubmissionID,StudentID,Problem,Status,Score,CodeLength,SubmissionTime
43629,7,4,Runtime Error,0,879,12:20:52
43628,31,3,Runtime Error,0,521,12:20:38
43627,106,2,Wrong Answer,0,10,12:20:27
43626,101,4,Wrong Answer,0,2330,12:20:27
43625,56,2,Wrong Answer,30,616,12:20:22
43624,13,2,Wrong Answer,0,1261,12:20:15
43623,84,2,Runtime Error,0,402,12:20:12
43622,78,2,Runtime Error,0,481,12:20:11
43621,31,3,Wrong Answer,0,521,12:20:11
43620,58,3,Wrong Answer,0,704,12:20:09
43619,46,2,Compile Error,0,1789,12:20:06
```

  – Student IDs are replaced by unique labels.

Programming for Business Computing

Making Histograms

Ling-Chieh Kung

Department of Information Management
National Taiwan University

Submission times

- We are interested in the students' **submission times**.
  - Is it true that students (altogether) make more submits when it is closer to the end of the exam?
- To answer this question, we may:
  - First, find the submission times, which is defined as the number of seconds since the exam starts (at 9:20:00) of a submission.
  - Second, draw a **histogram** for them: Number of submissions in [0, 1000), [1000, 2000), …, and [10000, 11000).

Data processing

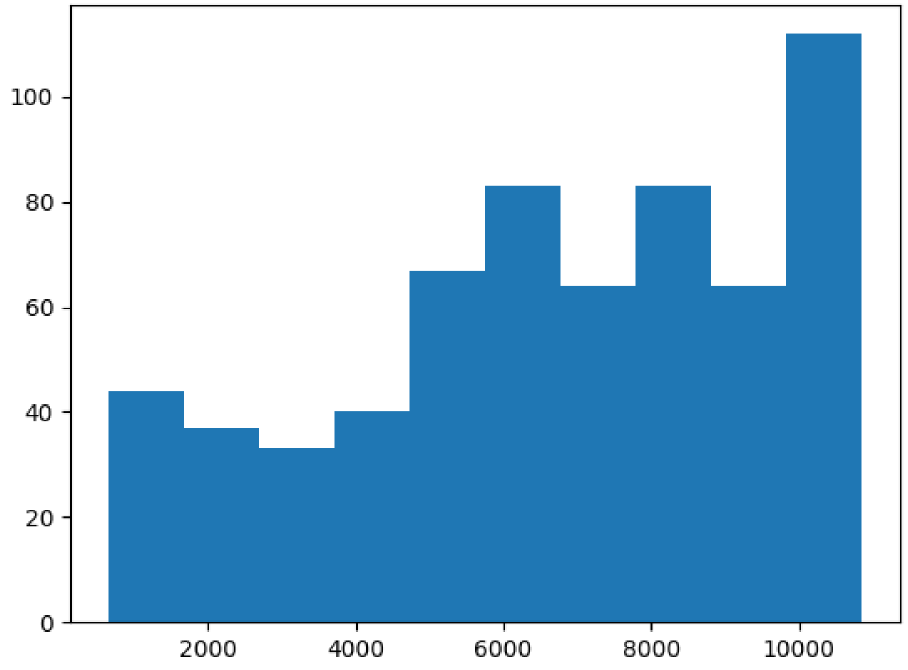
- First, we find the submission times:

```
import csv, datetime

def findSubTimes(fileName)
  fh = open(fileName, "r")
  csvFile = csv.DictReader(fh)

  subTimes = [] # to store submission times
  for row in csvFile:
    dt = datetime.datetime.strptime(row["SubmissionTime"], "%H:%M:%S").time()
    sub = (dt.hour - 9) * 3600 + (dt.minute - 20) * 60 + dt.second
    subTimes.append(sub)

  fh.close()
  return subTimes
```

Making a histogram

- Second, we make a histogram:

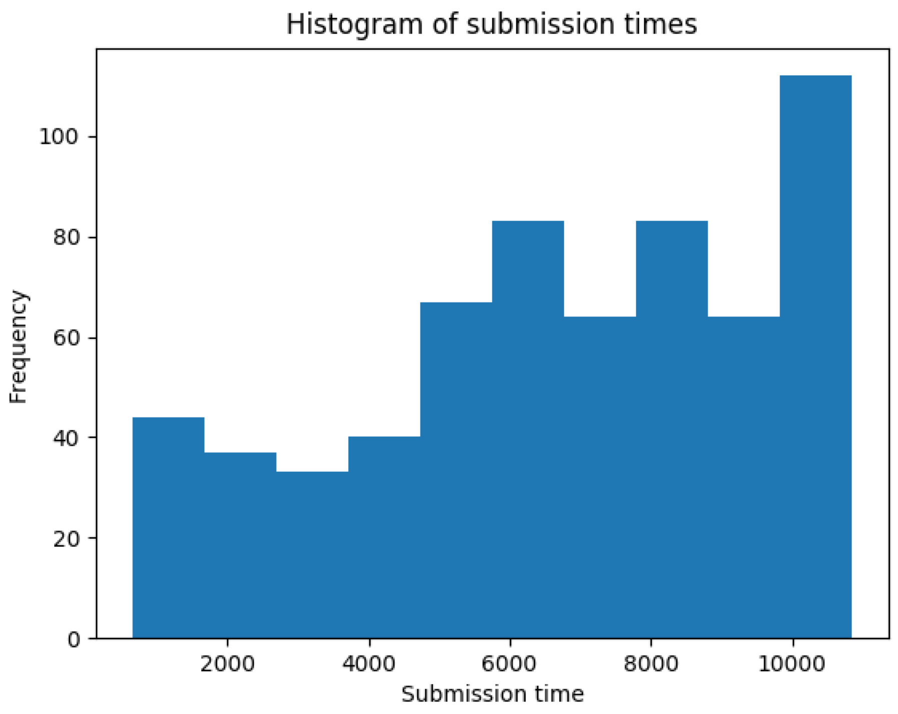
```
import matplotlib.pyplot as py

subTimes = findSubTimes("midterm2.csv")
# print(subTimes) # just testing

py.hist(subTimes)
py.show()
```



Decorating a histogram

- A plot should have *x*-label, *y*-label, and title (caption).

```
import matplotlib.pyplot as py

subTimes = findSubTimes("midterm2.csv")

py.hist(subTimes)
py.xlabel("Submission time")
py.ylabel("Frequency")
py.title("Histogram of submission times")
py.show()
```

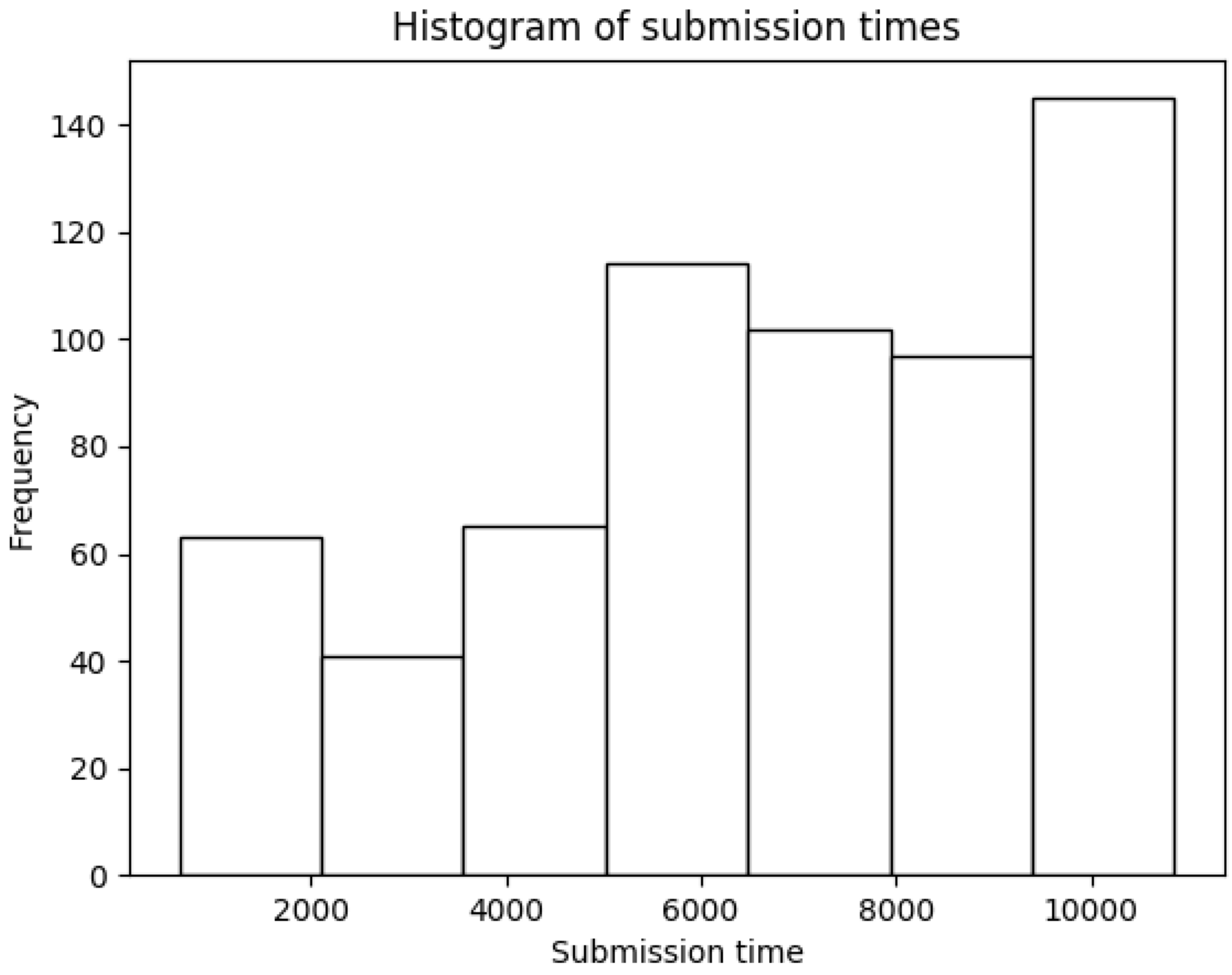


Setting up a histogram

- We may want to determine bar colors, edge colors, and the number of classes.

```
import matplotlib.pyplot as py

subTimes = findSubTimes("midterm2.csv")

py.hist(subTimes, bins = 7, color = "white", edgecolor = "black")
py.xlabel("Submission time")
py.ylabel("Frequency")
py.title("Histogram of submission times")
py.show()
```

Setting up a histogram

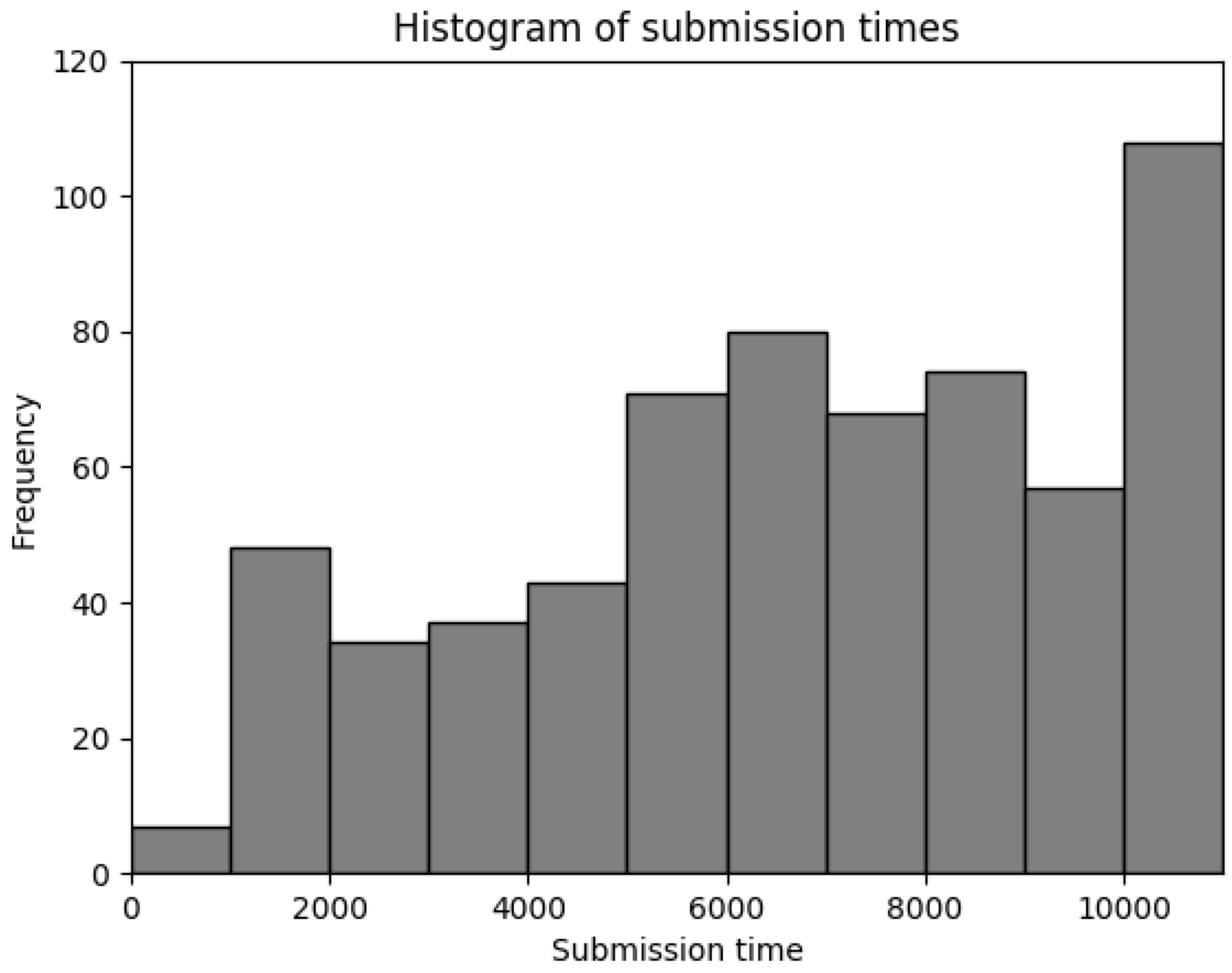


Setting up a histogram

- We may **specify all classes** by ourselves.

```
import matplotlib.pyplot as py

subTimes = findSubTimes("midterm2.csv")

endpoints = range(0, 12000, 1000)
py.hist(subTimes, bins = endpoints, facecolor = "gray", edgecolor = "black")
py.ylim(0, 120)
py.xlim(0, 11000)
py.xlabel('Submission time')
py.ylabel('Frequency')
py.title('Histogram of submission times')
py.show()
```

Setting up a histogram



Obtaining the frequencies

- We may want to obtain the frequencies (and/or class endpoints).

```
import matplotlib.pyplot as py

subTimes = findSubTimes("midterm2.csv")

n, bins, patches = py.hist(subTimes, bins = range(0, 12000, 1000))

print(n) # the frequencies
print(bins) # the class endpoints
```

Programming for Business Computing

Making Bar Charts and Pie Charts

Ling-Chieh Kung

Department of Information Management
National Taiwan University

Each student's number of submissions

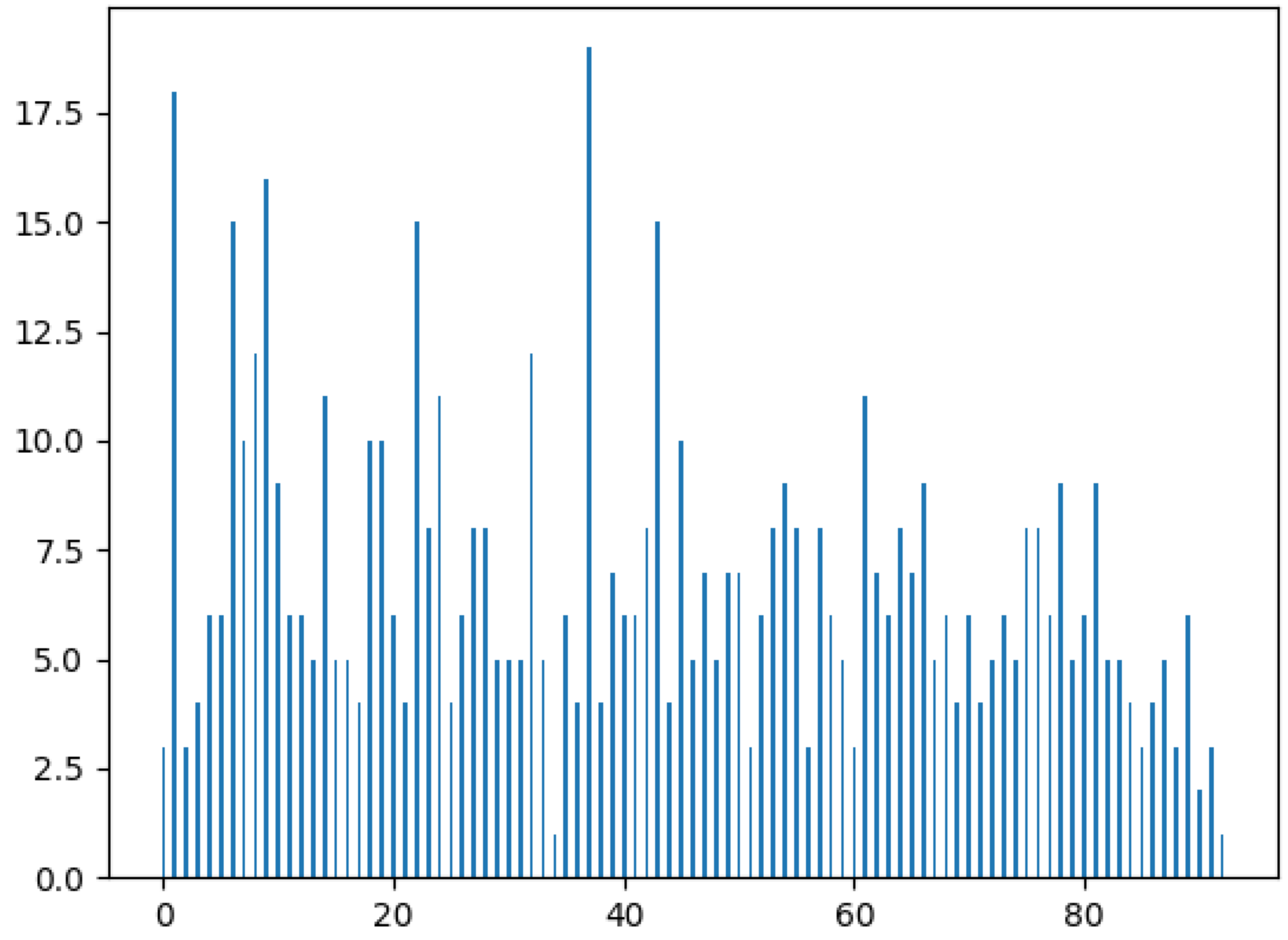
- Students may have different number of submissions.
- Beside using a histogram to visualize the distribution of the number of submissions, we may also use a **bar chart** to visualize all these numbers.

Making a bar chart

```
import csv, datetime

def findSubCnt(filePath):
  fh = open(filePath, "r")
  csvFile = csv.DictReader(fh)

  subCntDict = dict()

  for row in csvFile:
    sid = int(row["StudentID"])
    if sid in subCntDict:
      subCntDict[sid] += 1
    else:
      subCntDict[sid] = 1

  fh.close()

  return subCntDict
```

```
import matplotlib.pyplot as py

subCnt = findSubCnt("midterm2.csv")

seq = range(0, len(subCnt))
width = 0.35
py.bar(seq, subCnt.values(), width)
py.show()
```

Making a bar chart

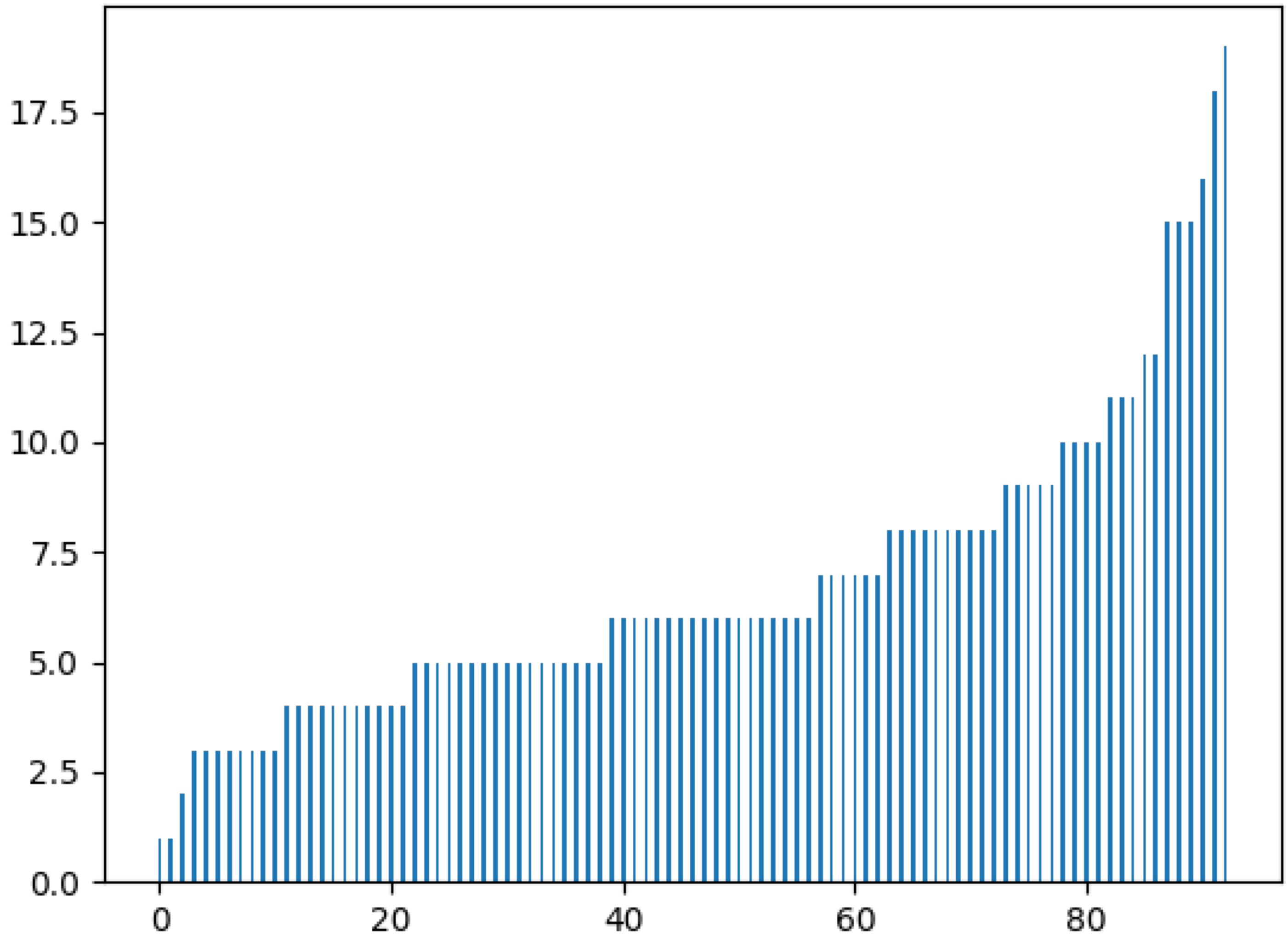


Sorting bars

```
import csv, datetime

def findSubCnt(filePath):
  fh = open(filePath, "r")
  csvFile = csv.DictReader(fh)

  subCntDict = dict()

  for row in csvFile:
    sid = int(row["StudentID"])
    if sid in subCntDict:
      subCntDict[sid] += 1
    else:
      subCntDict[sid] = 1

  fh.close()

  return subCntDict
```

```
import matplotlib.pyplot as py

subCnt = findSubCnt("midterm2.csv")

seq = range(0, len(subCnt))
width = 0.35
py.bar(seq, sorted(subCnt.values()), width)
py.show()
```

Sorting bars



Making a pie chart

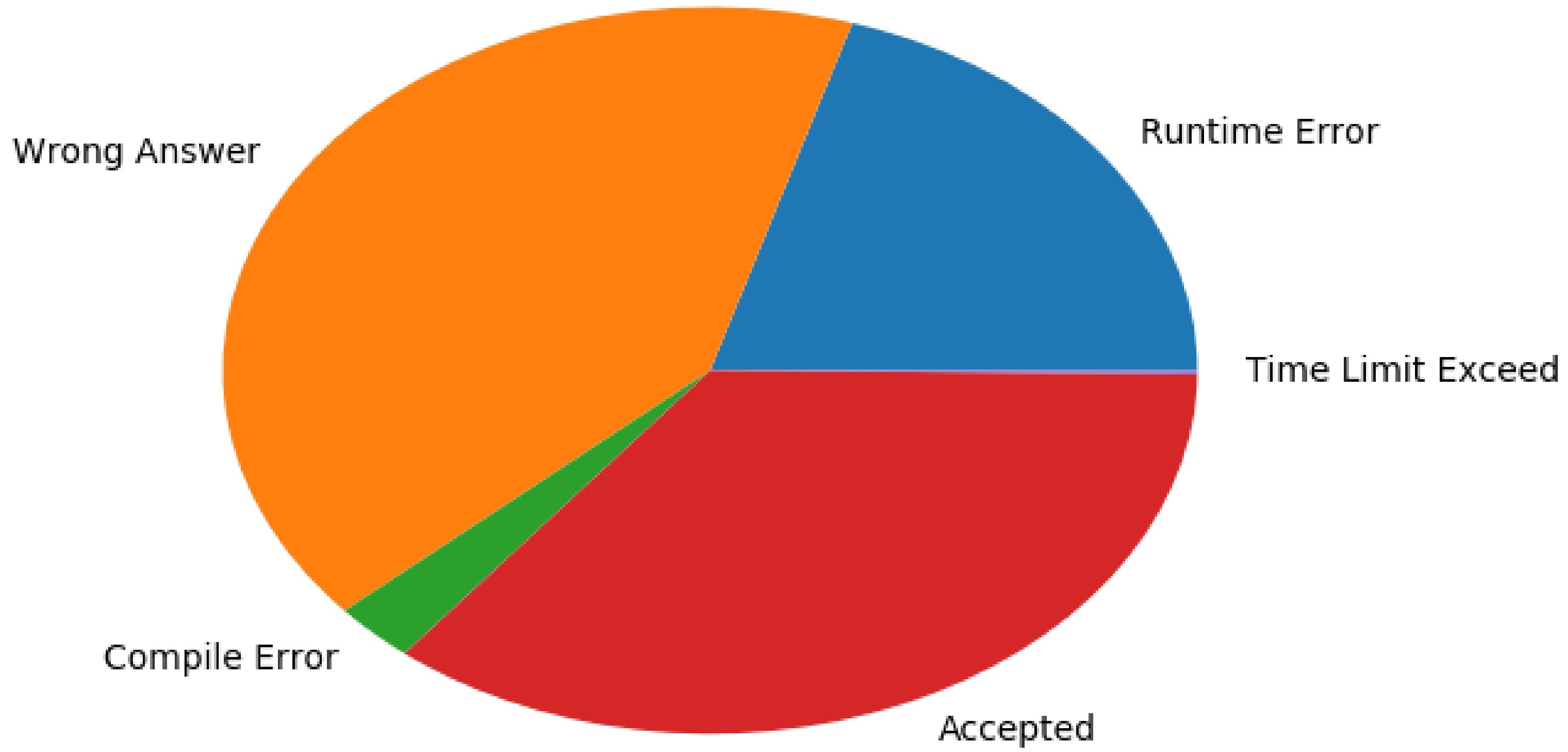
- When one submits, the result may be “Accepted”, “Wrong Answer”, “Runtime Error”, “Compile Error”, and “Time Limit Exceed”.
- We may use a **pie chart** to visualize their proportions.

Making a pie chart

```python
import csv, datetime

def findProp(filePath):
  fh = open(filePath, "r")
  csvFile = csv.DictReader(fh)

  propDict = dict()

  for row in csvFile:
    result = row["Status"]
    if result in propDict:
      propDict[result] += 1
    else:
      propDict[result] = 1

  fh.close()

  return propDict
```

```python
import matplotlib.pyplot as py

pf = findProp("midterm2.csv")

f = list(pf.values())
r = list(pf.keys())

py.pie(f, labels = r)
py.show()
```

Making a pie chart

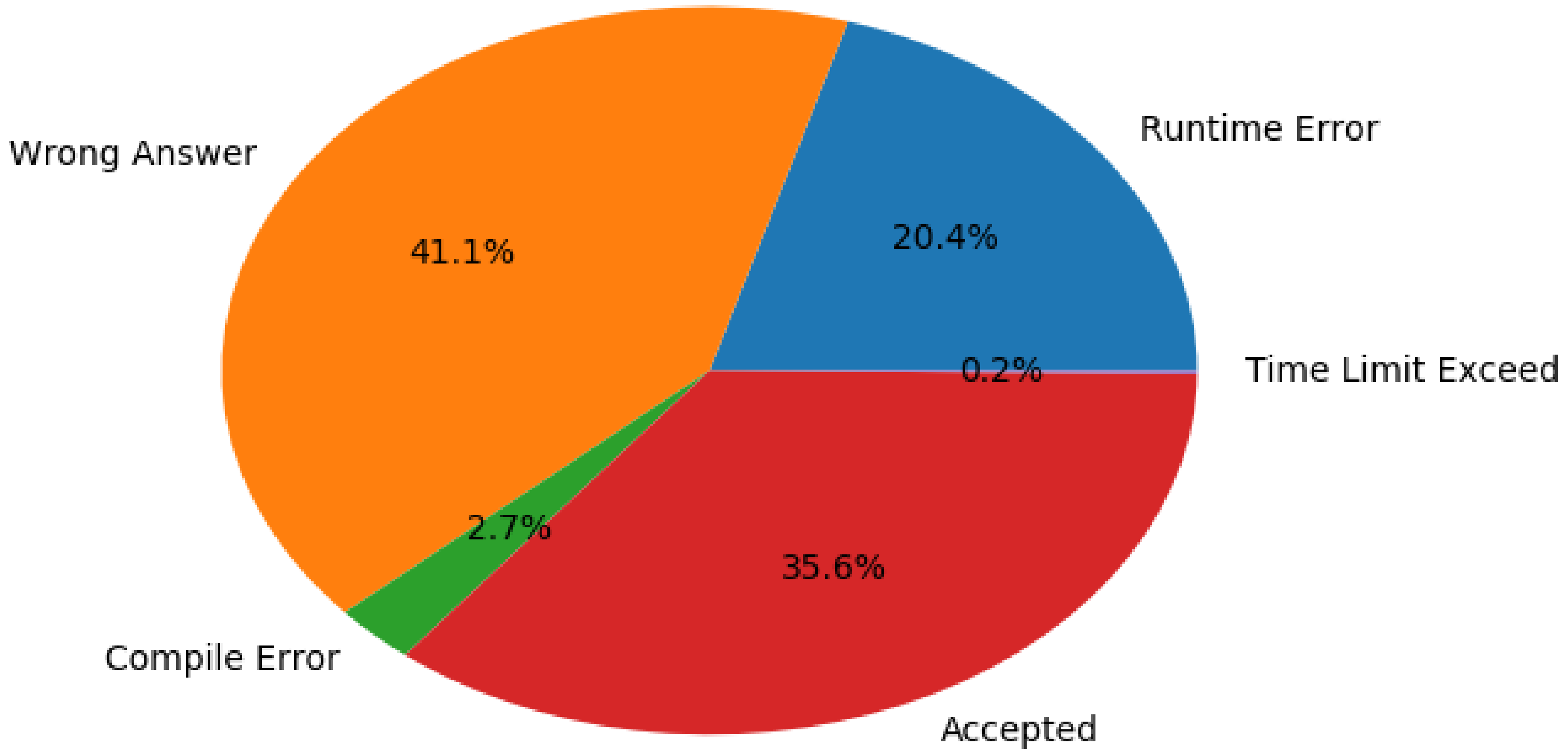


Adding data labels

```python
import csv, datetime

def findProp(filePath):
  fh = open(filePath, "r")
  csvFile = csv.DictReader(fh)

  propDict = dict()

  for row in csvFile:
    result = row["Status"]
    if result in propDict:
      propDict[result] += 1
    else:
      propDict[result] = 1

  fh.close()

  return propDict
```

```python
import matplotlib.pyplot as py

pf = findProp("midterm2.csv")

f = list(pf.values())
r = list(pf.keys())

py.pie(f, labels = r, autopct = "%1.1f%%")
py.show()
```

Making a pie chart



Programming for Business Computing

Making Scatter Plots and Line Charts

Ling-Chieh Kung

Department of Information Management

National Taiwan University

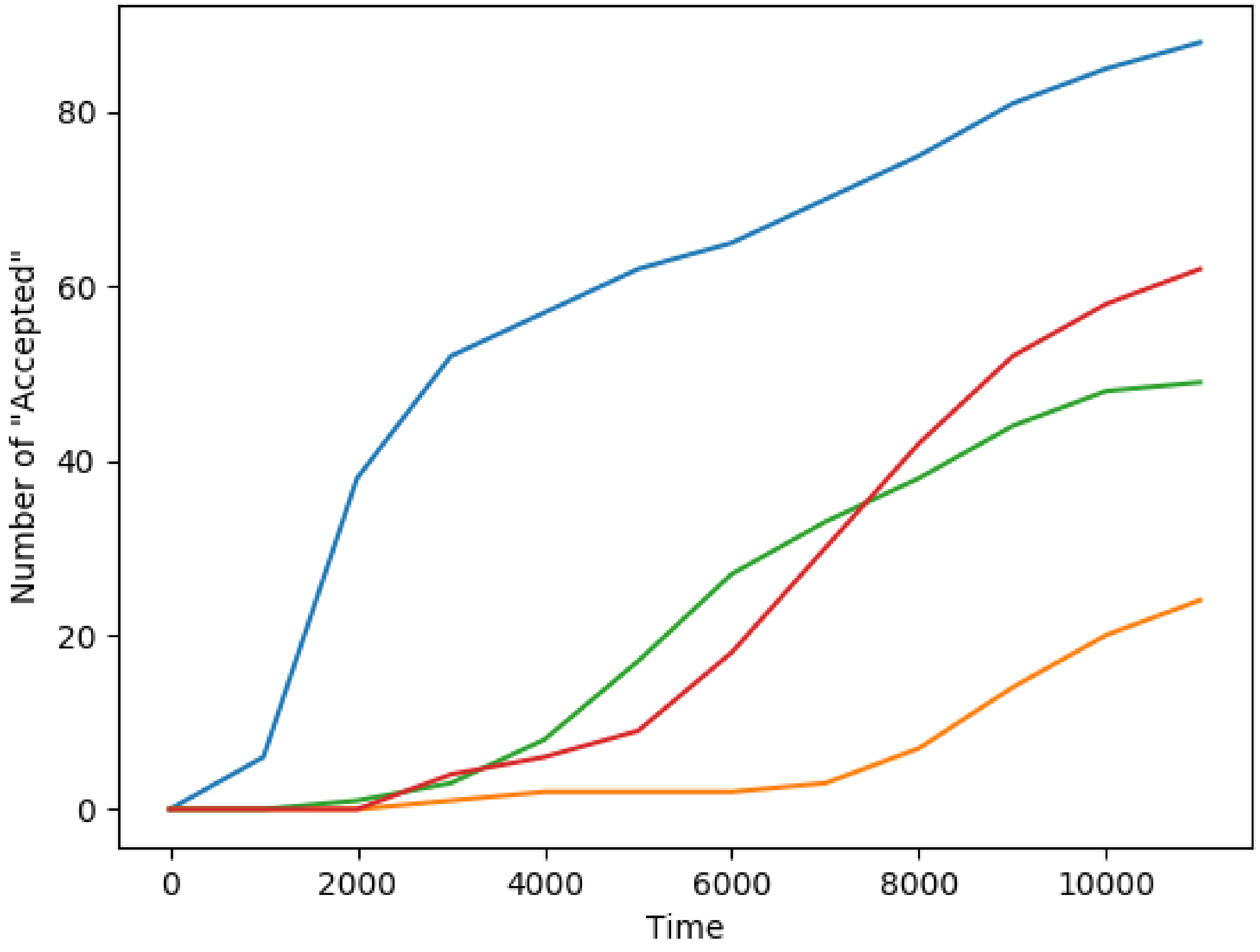
Road to AC

- How was each problem be answered?
  - The speed of getting "Accepted".
  - The difference between problems.
- We may draw a **line chart** and use four lines to represent the cumulative numbers of "Accepted" up to a certain time point, one for each problem.
- Note that this is not an easy task if MS Excel is the only tool!
  - How to process the data and calculate these numbers?

Making a line chart (attempt 1)

```
import matplotlib.pyplot as py

p1 = [0, 6, 38, 52, 57, 62, 65, 70, 75, 81, 85, 88]
p2 = [0, 0, 0, 1, 2, 2, 2, 3, 7, 14, 20, 24]
p3 = [0, 0, 1, 3, 8, 17, 27, 33, 38, 44, 48, 49]
p4 = [0, 0, 0, 4, 6, 9, 18, 30, 42, 52, 58, 62]
times = range(0, 12000, 1000)

py.plot(times, p1)
py.plot(times, p2)
py.plot(times, p3)
py.plot(times, p4)
py.xlabel('Time')
py.ylabel('Number of "Accepted"')
py.show()
```

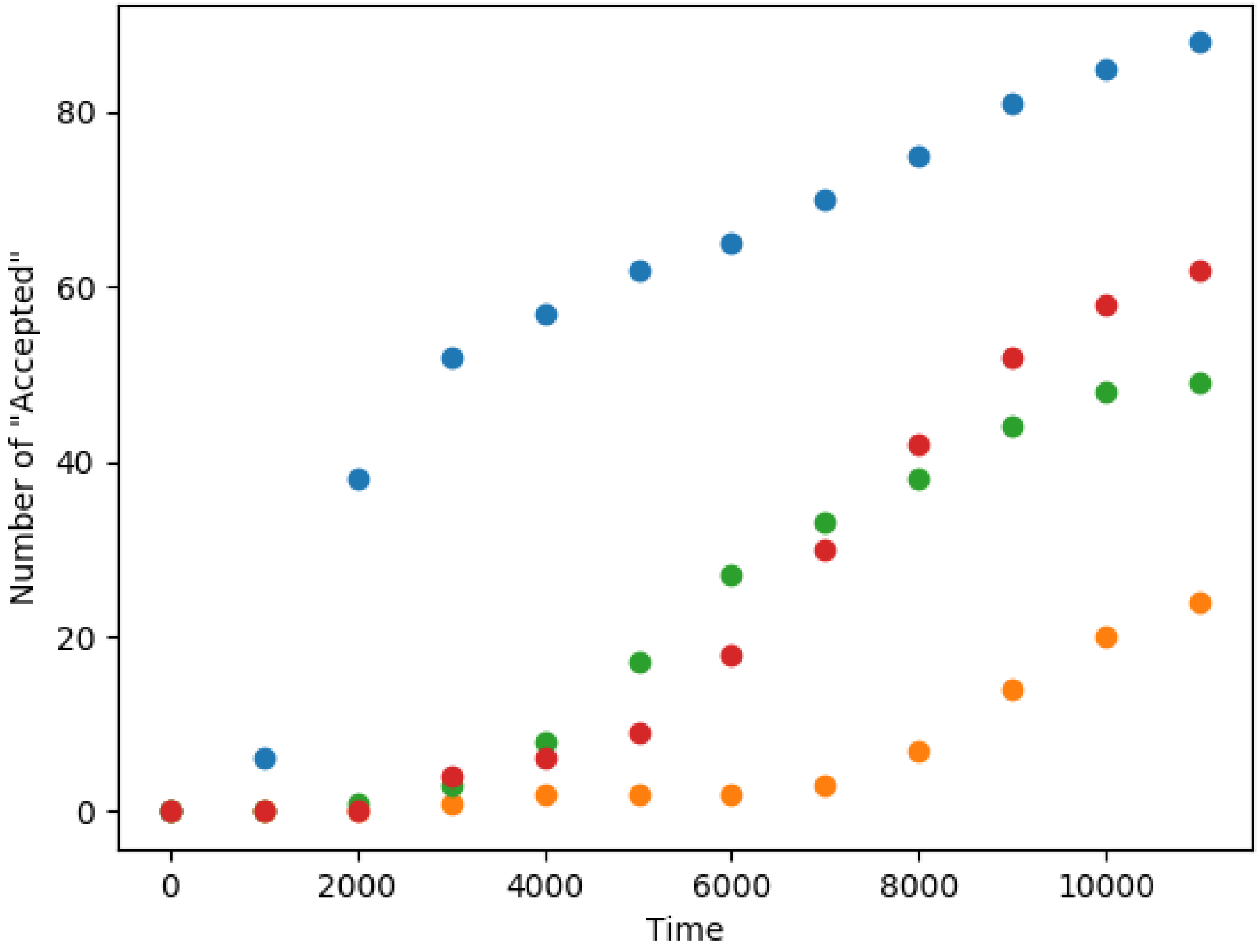
Making a line chart (attempt 1)



Making a line chart (attempt 2)

```
import matplotlib.pyplot as py

p1 = [0, 6, 38, 52, 57, 62, 65, 70, 75, 81, 85, 88]
p2 = [0, 0, 0, 1, 2, 2, 2, 3, 7, 14, 20, 24]
p3 = [0, 0, 1, 3, 8, 17, 27, 33, 38, 44, 48, 49]
p4 = [0, 0, 0, 4, 6, 9, 18, 30, 42, 52, 58, 62]
times = range(0, 12000, 1000)

py.plot(times, p1, 'o')
py.plot(times, p2, 'o')
py.plot(times, p3, 'o')
py.plot(times, p4, 'o')
py.xlabel('Time')
py.ylabel('Number of "Accepted"')
py.show()
```

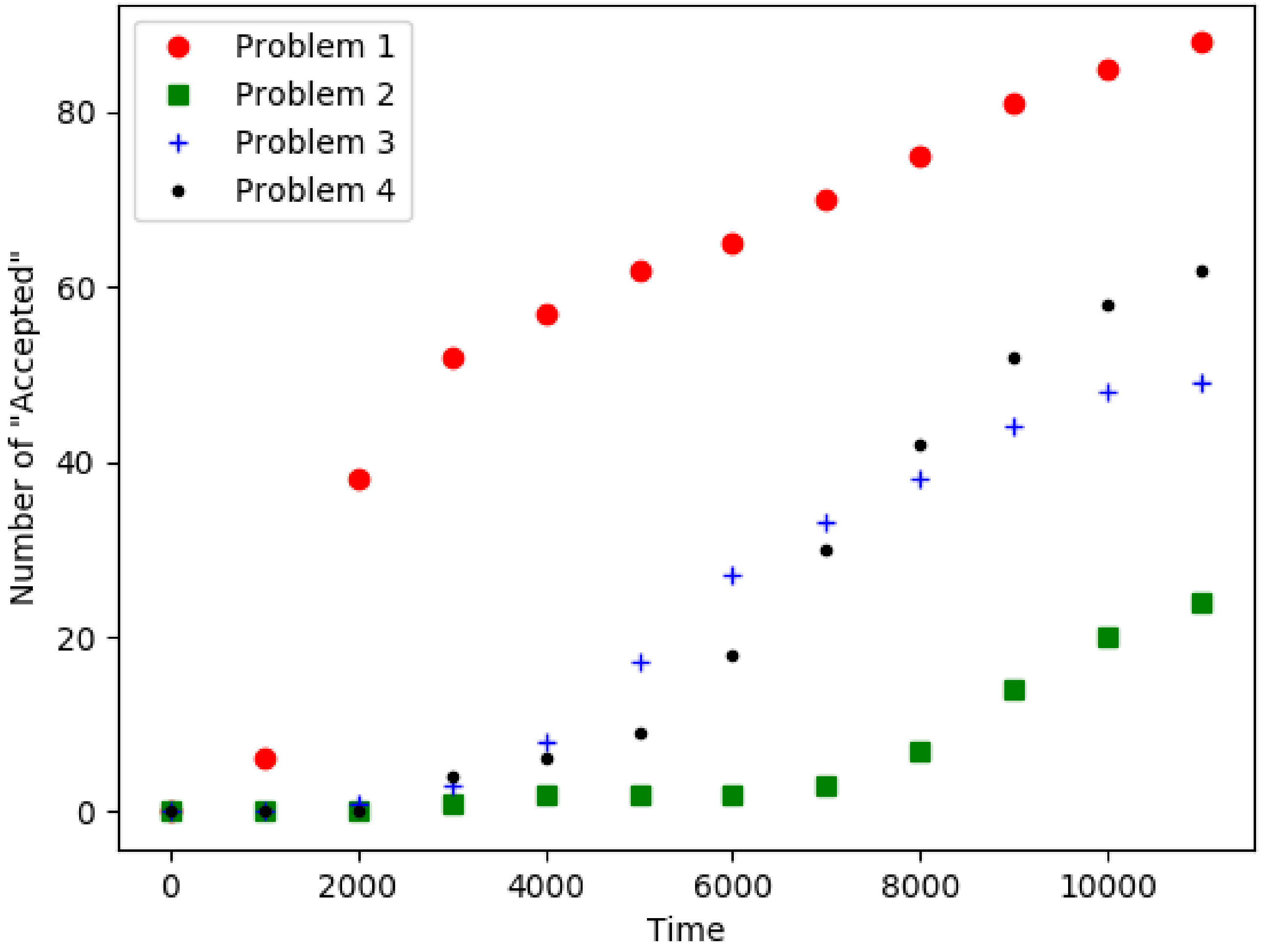
Making a line chart (attempt 2)



Making a line chart (attempt 3)

```
import matplotlib.pyplot as py

p1 = [0, 6, 38, 52, 57, 62, 65, 70, 75, 81, 85, 88]
p2 = [0, 0, 0, 1, 2, 2, 2, 3, 7, 14, 20, 24]
p3 = [0, 0, 1, 3, 8, 17, 27, 33, 38, 44, 48, 49]
p4 = [0, 0, 0, 4, 6, 9, 18, 30, 42, 52, 58, 62]

times = range(0, 12000, 1000)
py.plot(times, p1, 'ro', label = "Problem 1")
py.plot(times, p2, 'gs', label = "Problem 2")
py.plot(times, p3, 'b+', label = "Problem 3")
py.plot(times, p4, 'k.', label = "Problem 4")
py.legend(loc = 'upper left')
py.xlabel('Time')
py.ylabel('Number of "Accepted"')
py.show()
```

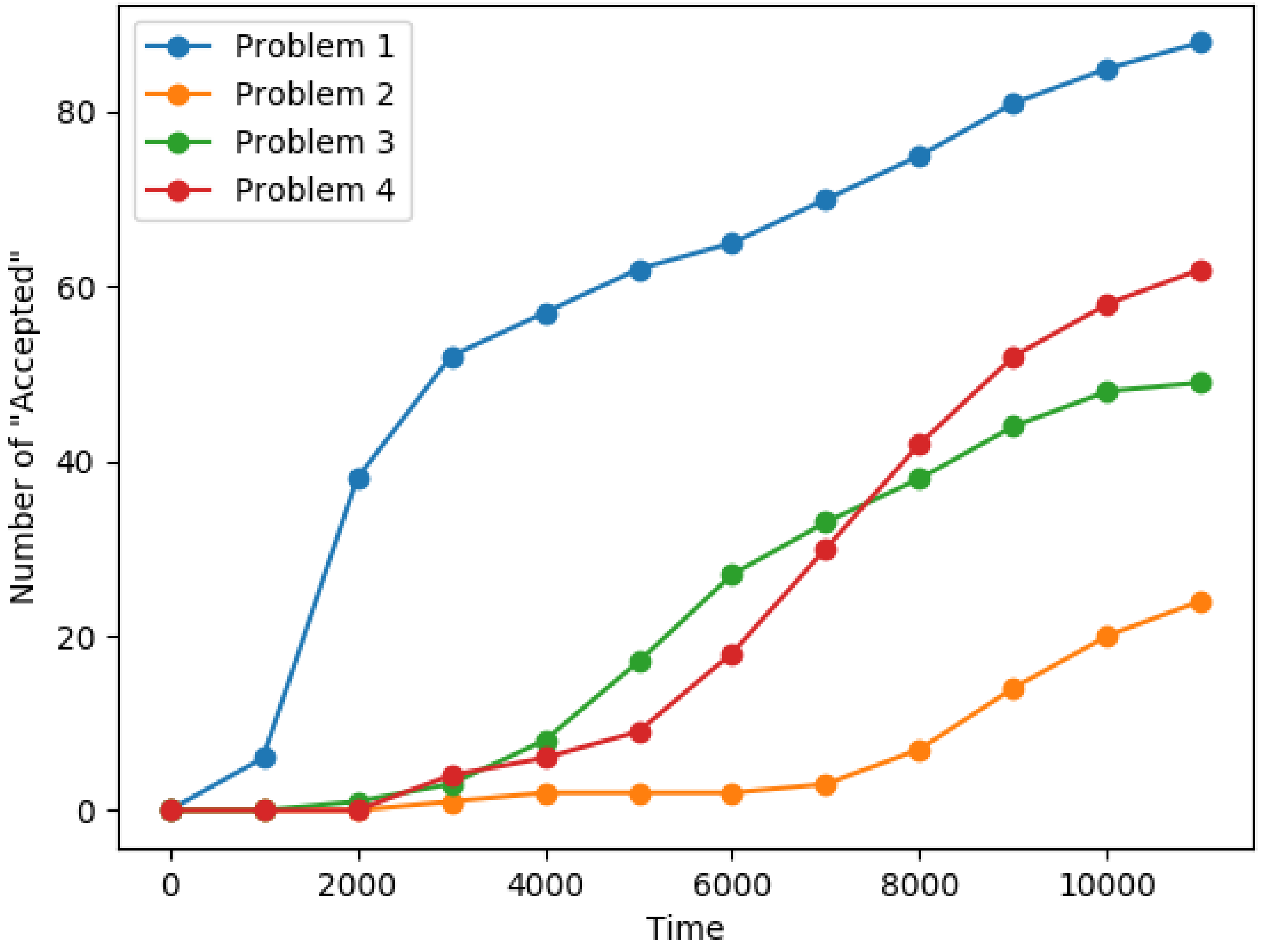
Making a line chart (attempt 3)



Making a line chart (attempt 4)

```
import matplotlib.pyplot as py
p1 = [0, 6, 38, 52, 57, 62, 65, 70, 75, 81, 85, 88]
p2 = [0, 0, 0, 1, 2, 2, 2, 3, 7, 14, 20, 24]
p3 = [0, 0, 1, 3, 8, 17, 27, 33, 38, 44, 48, 49]
p4 = [0, 0, 0, 4, 6, 9, 18, 30, 42, 52, 58, 62]

times = range(0, 12000, 1000)
py.plot(times, p1, label = "Problem 1", marker = 'o')
py.plot(times, p2, label = "Problem 2", marker = 'o')
py.plot(times, p3, label = "Problem 3", marker = 'o')
py.plot(times, p4, label = "Problem 4", marker = 'o')
py.legend(loc = 'upper left')
py.xlabel('Time')
py.ylabel('Number of "Accepted"')
py.show()
```

Making a line chart (attempt 4)



Summary

- Use classes to organize and modularize your program.
  - Comments are important!
- Process data and visualize them with libraries.
  - Almost everything you want have been implemented and put somewhere on the Internet.
  - Search, copy, modify, try, understand, and create!