Assignment #1
Turtle Charter

Digital Computer Concept and Practice
dccp@hcil.snu.ac.kr

1 Introduction

이번 과제는 지난 과제 'Turtle로 라이언 그려보기'를 바탕으로 익숙해진 Python Turtle Library를 사용해 데이터를 받아와 그래프를 그려보는 과제입니다. 3 종류의 그래프 (Bar chart, Scatterplot, Line chart)를 Turtle로 그리게 될 것이며 이를 통해 단순한 형태의 데이터 시각화를 경험하는 것을 목표로 합니다. 이번 과제의 결과물은 다음 영상과 같은 형태를 따라야 합니다. (YouTube)

- 제출 기한: 10/07 수요일 **23:59**
- 제출 방법:
  - eTL 과제란에 **소스 코드**(`chart.py` 파일)와 **각각의 그래프** 캡쳐본을 제출
    - `2020-23456-오영택.py` (기존 `chart.py`)
    - `2020-23456-오영택-bar.jpeg`
    - `2020-23456-오영택-scatter.jpeg`
    - `2020-23456-오영택-line.jpeg`
  - 모든 제출 파일을 학번-이름`.zip`의 **압축 파일 형식**으로 묶어 제출해주시길 바랍니다.
    - `2020-34567-김수현.zip`

2 Explanation

2.1 Implementation

- 제공해드리는 skeleton code 파일은 데이터 `data.txt`, 그리고 두 개의 소스코드로 이뤄졌습니다. 소스코드는 각각 실행파일인 `drawing.py`와 여러분이 구현해야 할 `chart.py`로 나누어져 있습니다.
- 프로그램은 다음과 같은 명령어를 통해 실행됩니다.
  - `python3 drawing.py bar`
  - `python3 drawing.py scatter`
  - `python3 drawing.py line`

  각각의 명령어는 3번째 argument에 해당하는 차트를 그리게 됩니다. 여러분은 **chart.py**의 Skeleton code 상에서 아래와 같이 축과 그래프를 그리는 함수를 채워 구현해주시면 됩니다.
- 구현해야 할 함수는 다음과 같습니다.

- `def axis()` : $x > 0$, $y > 0$ 사분면만을 표현하는 두 axis를 그림
- `def bar_chart()` : bar chart를 그림
- `def scatterplot()` : scatterplot을 그림
- `def line_chart()` : line chart를 그림

구현하신 함수는 `drawing.py`가 실행될 때 argument에 따라 자동으로 실행됩니다. 예를 들어, `python3 drawing.py scatter`를 실행하실 경우 축을 그리는 `axis()`와 scatterplot을 그리는 `scatterplot()` 함수가 차례로 실행됩니다. 잘못된 argument를 입력하실 경우 `axis()`만 실행되고 프로그램이 멈추게 됩니다.

각각의 그래프에 대한 상세한 설명은 다음과 같습니다.

Bar Chart

Bar Chart란 표현 값에 비례한 높이를 지닌 직사각형 막대로 범주형 데이터를 표현하는 그래프입니다. 이번 프로젝트에서는 수직 막대 그래프만 다루게 됩니다. Turtle로 그릴 Bar chart는 다음과 같은 조건을 만족해야 합니다.

- 각각의 Bar는 각 label과 대응해야 합니다. 다시 말해, bar의 개수는 label의 개수와 같습니다.
- Bar은 label과 대응하는 지정된 색으로 칠해져야 합니다.
- Bar의 높이는 각 label의 element 개수에 비례해야 합니다. A label의 element 수가 50개, B label의 element 수개 100개라면 B의 높이는 A의 2배가 되어야 합니다.
- Bar의 너비는 일정해야 합니다.

Scatterplot

Scatterplot은 이름처럼 각각의 데이터 포인트들을 흩뿌려놓은 형태로 x축과 y축으로 이뤄진 2차원의 공간에 데이터를 표현합니다. Turtle로 그릴 scatterplot는 다음과 같은 조건을 만족해야 합니다.

- 각각의 element는 속이 차 있는 **단색**의 circle로 표현되어야 합니다.
- element들의 색깔은 해당하는 label의 색으로 지정되어야 합니다. (ex) A label: "red")
- 각각의 element의 x좌표 순서는 `LABELS` constant의 index 순서와 같아야 합니다.
- 각 label에 대응하는 x좌표 사이의 간격은 일정해야 합니다.
- 각각의 element의 y좌표는 element의 value에 비례해야 합니다. 예를 들어 A label의 특정 element가 70의 value를 가졌는데 (50, 140)에 위치해 있다면, A label의 value 80을 가지는 다른 element는 (50, 160)에 위치해야 합니다.

Line Chart

Line Chart는 line segment로 연결된 '마커'(marker)라는 이름의 일련의 데이터 지점으로 정보를 표시하는 차트입니다. Turtle로 그릴 line chart는 다음과 같은 조건을 만족해야 합니다.

- label당 marker는 하나입니다.
- marker는 label과 대응하는 지정된 색으로 칠해져야 하고, 단색의 circle 모양을 가져야 합니다.

- 연속된 marker는 검은색 line segment로 연결되어 있어야 합니다.
- marker의 x좌표 순서는 `LABELS` constant의 index 순서와 같아야 합니다.
- 각 label에 대응하는 marker의 x좌표 사이 간격은 일정해야 합니다.
- marker의 y좌표는 각 label의 element 개수에 비례해야 합니다.

Constants

Skeleton code 안에 정의되어 있는 constant들에 대한 설명은 다음과 같습니다. 엄격한 디자인 가이드라인이 없기에 굳이 사용하실 필요는 없으나, 쉬운 구현을 위해 (몇 가지 제약 조건을 쉽게 만족시키기 위해) 사용하는 것을 추천드립니다.

- `AXIS_LENGTH` : 축의 길이, x, y축 모두 동일한 값을 가집니다.
- `AXIS_MARGIN`: 축에 화살표를 그리기 위한 MARGIN을 나타냅니다. 결과적으로 축의 길이는 `AXIS_LENGTH + AXIS_MARGIN`이 되고 끝에 `AXIS_MARGIN` 만큼의 공간을 차지하는 화살표를 그려야 합니다. 화살표의 모양, 크기에 대해 정해진 사항은 없으니 굳이 활용하지 않으셔도 됩니다.
- `LABELS`: data의 label을 나타냅니다. 이번 과제에서 제공하는 dataset의 label은 A ∼ F까지입니다.
- `COLORS`: 각 label과 대응하는 색깔을 나타냅니다. 예를 들어 `LABELS[3]`은 `COLORS[3]`과 대응합니다.
- `PIXEL_PER_VAL`: `bar_chart`, `scatterplot`, `line_chart`에서 local하게 쓰이는 constant 입니다. value가 실제 공간에 몇 pixel로 그려질지를 지정합니다. 예를 들어 `bar_chart`에서 특정 val의 value가 `50`이라면, 그 bar의 실제 높이는 `50 * PIXEL_PER_VALUE`가 되어야 합니다. 다시 말해, turtle 프로그램 상에서 그만큼 거리가 떨어져 있어야 합니다.

2.2 Background

이 과제는 수업시간에 다루지 않은 내용을 일부 포함하고 있어 다음과 같이 추가 설명을 첨부합니다.

File read

- 여태껏 우리는 무엇인가 **입력** 받을 때 사용자가 직접 키보드로 입력하는 방식을 사용했습니다. 하지만 이번 과제와 같이 많은 데이터를 한 번에 입력을 받고 싶을 경우 파일에서 입력을 받아 배열에 저장한다면 더욱 효율적으로 이들을 관리할 수 있습니다.
- 파일을 생성하기 위해 우리는 파이썬 내장 함수 `open`을 사용합니다. 인수로 **'파일 이름'**과 **'파일 열기 모드'**를 입력해주면 파일이 읽기모드로 열 수 있습니다.
- 다음 코드를 통해 파일이 담고 있는 내용을 `file_str` 변수에 string 형태로 저장할 수 있습니다.

```
f = open("./data.txt", "r")
file_str = f.read()
```

- file을 다 사용하신 후 `f.close()`를 호출해 파일을 닫아야 합니다.
- 인터넷 검색을 통해 더 쉬운 방법을 찾으셨다면, 그 방법을 쓰셔도 무방합니다.

List & List Indexing

- python에서 데이터를 관리하는 가장 쉬운 방법 중 하나는 list를 사용하는 것입니다.
- list는 여러 개의 데이터를 모아 순서대로 정리해주는 데이터 형식입니다.
- 생성된 list를 **indexing**을 통해 접근 / 조작할 수 있습니다.
- 예를 들어, 대괄호([]) 안에 index를 넣어주면 list의 해당 번째에 있는 원소를 반환해줍니다. 단, list의 첫 번째 element의 index는 `0` 이고, 길이가 `n`인 list의 마지막 index는 `n-1`입니다. 또한, 끝에서 n번째 element에 접근하기 위해서는 `-n` 값을 index로 넣어주면 됩니다.

```
arr = [‘1‘, ‘2‘, ‘3‘, ‘A‘, ‘B‘]       # list declaration
print(‘first element: ‘, arr[0])      # first element: 1
print(‘last element : ‘, arr[4])      # last element : B
print(‘last element : ‘, arr[-1])     # last element : B
```

String split

- `data.txt` 안에 있는 데이터는 string 형식으로 저장되게 됩니다.
- 이 연속적인 데이터를 처리하기 위해서는 이를 잘 나누어 list 형태로 저장해야 합니다.
- 이를 위해 **`split`**이라는 함수를 사용할 수 있습니다. 기본적으로 주어진 string을 스페이스(공백)를 기준으로 분리하여 `list`에 저장하는 기능을 수행합니다.
- 예를 들어, 다음 코드를 실행하면 arr에는 `‘1’, ‘2’, ‘3’, ‘A’, ‘B’`를 가진 list가 저장됩니다.

```
arr = ‘1 2 3 A B‘.split()
```

- 다른 구분자를 사용하고 싶다면, 그 문자를 `split` 함수에 argument로 주면 됩니다.
- 예를 들어, 다음 코드를 실행하면 arr에는 `‘Hell‘, ‘W‘, ‘rld’`를 가진 list가 저장됩니다.

```
arr = ‘HelloWorld‘.split(‘o‘)
```

3 Cautions

- **Grading Policy**
  - 위에서 이야기했듯이, YouTube 동영상에 나온 3가지 차트와 비슷한 모양이면 됩니다.
  - **영상을 꼭 참고해서 그려주세요.**
  - 엄격한 디자인 가이드라인은 없지만, 다음과 같은 조건들을 지켜야 합니다.
    - x축, y축 공간을 90% 이상 활용하셔야 합니다.
    - ex) bar chart는 axis 길이가 500이면 bar의 최대 높이가 450 이상이어야 합니다.
    - x축, y축 길이를 내부 element가 넘어가면 안 됩니다.
    - 각각의 graph element는 label과 대응되는 색깔을 정확히 표시해야 합니다.
    - 그 외 위에서 명시한 각 chart의 조건들을 지켜야 합니다.
  - 이번 과제에서 채점은 제공해 드린 `dataset.txt`만을 기반으로 진행됩니다.

- **Late Policy**
  - 연장 제출은 **5일**까지 허용됩니다. (9/28 23:59까지)
  - 연장 제출시 하루 **20%**씩 감점됩니다. (하루 20%, 이틀 40%···)
- python 3.8.5 인터프리터로 소스코드가 실행되지 않을 시 **0점 처리**됩니다.
- 문의사항은 eTL 질의응답 게시판 혹은 조교 메일로 보내주시기 바랍니다.

4 References

- UW canvas as3
- 데이터 시각화
- Wikipedia