

---

# PORTFOLIO

이름	김 혜 련
생년월일	1990. 06. 19
이메일	<a href="mailto:90x619@gmail.com">90x619@gmail.com</a>
전화번호	010-2608-9019
주소(GitHub)	github.com/KIMHYERYUN

---

# PORTFOLIO

## CONTENTS

- |     |  |   |
|-----|--|---|
| 01. | 연령별_인구수 데이터 분석   | 2012, 2022년 데이터를 통해 데이터를 불러오고 비교                              |
| 02. | 출생아 수 및 합계 출산율 분석  | 2012 ~ 2020년 데이터를 통해 데이터를 불러오고 추세파악                           |
| 03. | 서울시 구별 범죄현황 분석   | 2016년 구별 범죄 데이터와 서울시 CCTV 현황을 불러오고<br>데이터 전처리 및 시각화하여 상관관계 파악 |
| 04. | 소규모 Program<br>: Timer / Password Manager /<br>Sneak Game / Ping Pong Game | 개인 사이드 프로젝트로 Python 사용 경험 및 숙련도 향상을 위하여 진행                    |
| 05. | 기타   | 지속적으로 개인 공부 진행 중(Python 및 데이터 분석 위주 등)                        |

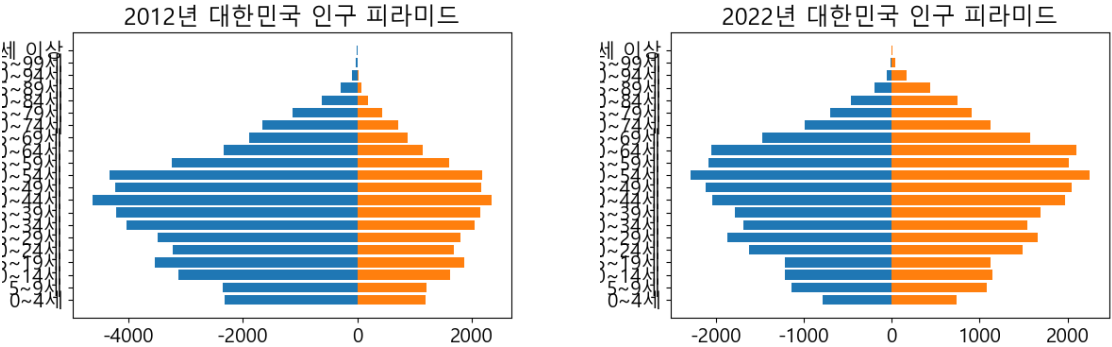
Project

# 연령별 인구 수 분석

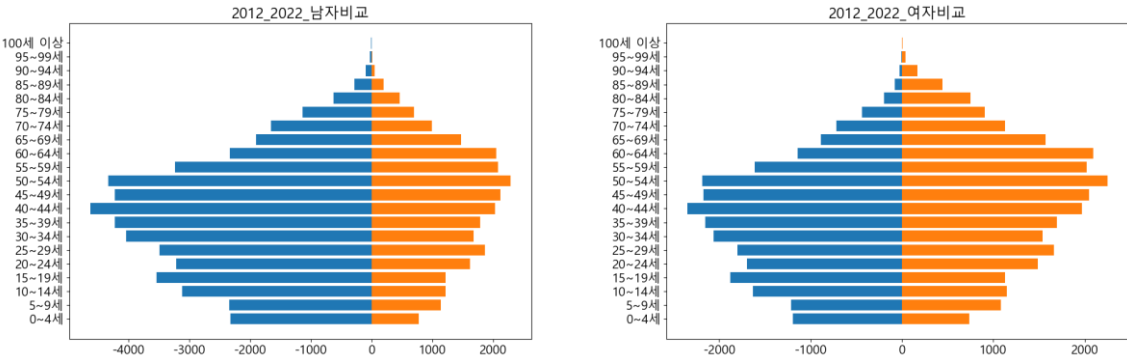
About project

2012, 2022년 데이터를 통해 데이터를 불러오고 비교

연도기준 성별 인구 수 그래프(좌-2012년, 우-2022년)



성별기준 연도별 인구 수 그래프(좌-남자, 우-여자)



Introduce project

작업 기간	2022. 05.
인력 구성(기여도)	BE 1명 / FE 1명 (FE 기여도 100%)
프로젝트 목적	2012, 2022년 데이터를 통해 데이터를 불러오고 비교
프로젝트 내용	연령별 인구수 분석을 위한 데이터 수집 2012, 2022년 / 남, 여 데이터 추출 2022년 인구 수가 2012년에 비해 변화된 양상 확인 - 시각화
주요 업무 및 상세 역할	1) 데이터 수집 2) 데이터 추출을 위한 방법 모색 3) 비교를 위한 시각화 그래프
사용언어 및 개발 환경	Python, Jupyter Notebook, VSC
참고 자료	Github 링크

Main work

데이터 추출 – 비교를 위한 데이터

- 기능 소개
  - 다양한 그래프 형태를 통한 비교
- 작업 내용
  - 데이터 추출 및 변환작업
  - 비교를 위한 컬럼명 통일

실행 및 디버그: 실행

디버그하거나 실행할 수 있는 파일 열기입니다.

실행 및 디버그

실행 및 디버그를 사용자 지정하려면 플러그를 열고 launch.json 파일을 만듭니다.

모든 자동 디버그 구성 표시.

Population\_Age.ipynb

C: > Users > 90x61 > projects > Population\_Age > Population\_Age.ipynb > M+연령별 인구수 분석 > M+남자 데이터 정의 > M+2011년

+ 코드 + Markdown

2022년 남자, 여자 비교

```
#연령 별 남자 데이터 - 수평 막대 그래프

#그래프 크기 변경
plt.figure(figsize=(10, 7))

#단위 : 천명으로 변환
plt.barh(df_m_2022.columns, df_m_2022.iloc[0] // 1000)

#연령 별 여자 데이터 - 수평 막대 그래프

#그래프 크기 변경
plt.figure(figsize=(10, 7))

#단위 : 천명으로 변환
plt.barh(df_w_2022.columns, df_w_2022.iloc[0] // 1000)

plt.barh(df_m_2022.columns, df_m_2022.iloc[0] // 1000)
plt.barh(df_w_2022.columns, df_w_2022.iloc[0] // 1000)
plt.show()

#대칭적으로 보여주는 그래프 그리기
plt.barh(df_m_2022.columns, -df_m_2022.iloc[0] // 1000)
plt.barh(df_w_2022.columns, df_w_2022.iloc[0] // 1000)

#타이틀
plt.title('2022년 대한민국 인구 피라미드')
#오류 : show 하기 전 저장 필수!!
plt.savefig('2022_인구피라미드.png', dpi=100)
plt.show()
```

...

100세 이상  
95~99세

Population\_Age.ipynb

C: > Users > 90x61 > projects > Population\_Age > Population\_Age.ipynb > M+연령별 인구수 분석 > M+남자 데이터 정의 > M+2011년

+ 코드 + Markdown

df\_m\_2022.columns

... Index(['0~4세 ', '5~9세 ', '10~14세 ', '15~19세 ', '20~24세 ', '25~29세 ', '30~34세 ', '35~39세 ', '40~44세 ', '45~49세 ', '50~54세 ', '55~59세 ', '60~64세 ', '65~69세 ', '70~74세 ', '75~79세 ', '80~84세 ', '85~89세 ', '90~94세 ', '95~99세 ', '100세 이상 '], dtype='object')

df\_w\_2022.columns

... Index(['0~4세 .1', '5~9세 .1', '10~14세 .1', '15~19세 .1', '20~24세 .1', '25~29세 .1', '30~34세 .1', '35~39세 .1', '40~44세 .1', '45~49세 .1', '50~54세 .1', '55~59세 .1', '60~64세 .1', '65~69세 .1', '70~74세 .1', '75~79세 .1', '80~84세 .1', '85~89세 .1', '90~94세 .1', '95~99세 .1', '100세 이상 .1'], dtype='object')

# 컬럼명 통일  
df\_w\_2022.columns = df\_m\_2022.columns  
df\_w\_2022.columns

... Index(['0~4세 ', '5~9세 ', '10~14세 ', '15~19세 ', '20~24세 ', '25~29세 ', '30~34세 ', '35~39세 ', '40~44세 ', '45~49세 ', '50~54세 ', '55~59세 ', '60~64세 ', '65~69세 ', '70~74세 ', '75~79세 ', '80~84세 ', '85~89세 ', '90~94세 ', '95~99세 ', '100세 이상 '], dtype='object')

#전국에 해당하는 연령별 인구수 추출  
#1. 문자열로 되어있는 숫자 -> 데이터 유형 변경 : 콤마 삭제, int로 변경 -> 반드시 저장  
df\_w\_2022.iloc[0] = df\_w\_2022.iloc[0].str.replace(',', '').astype(int)

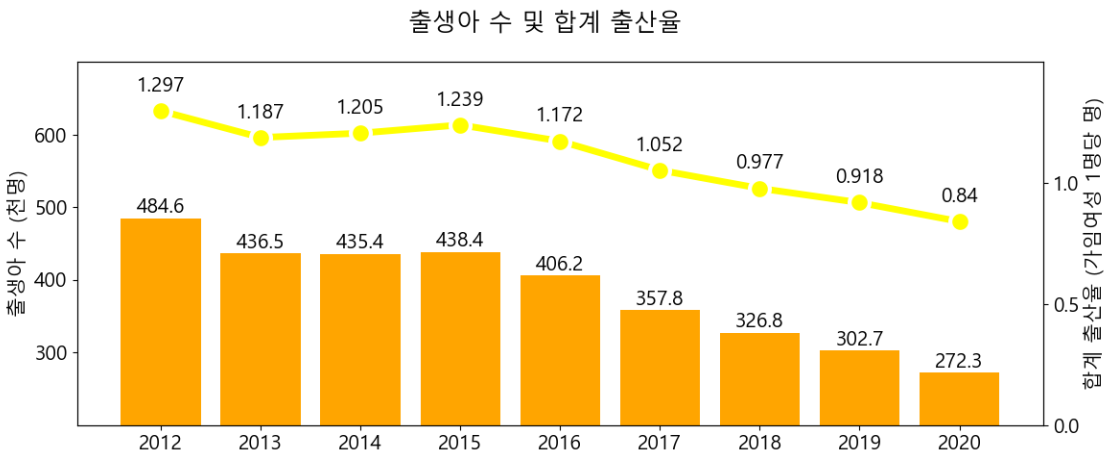
Project 02.

# 출생아 수 및 합계 출산율 비교

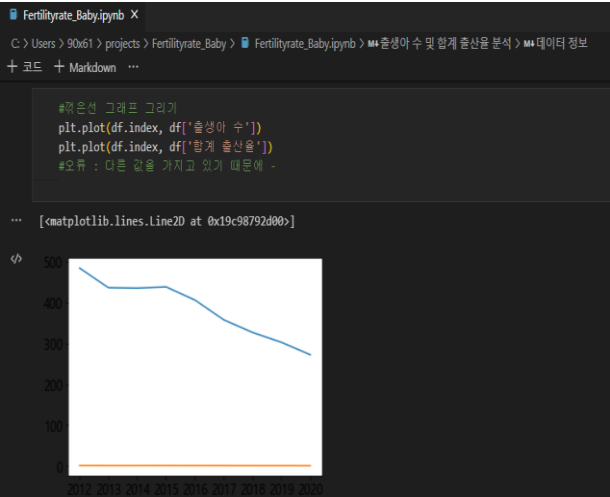
About project

2012 ~ 2020년 데이터를 통해 데이터를 불러오고 추세파악

연도별 출생아 수 및 합계 출산율 변화 양상



그래프 그리기 위한 데이터 사전작업 - 오류수정



Introduce project

작업 기간	2022. 05.
인력 구성(기여도)	BE 1명 / FE 1명 (FE 기여도 100%)
프로젝트 목적	2012 ~ 2020년 데이터를 통해 불러오고 추세 비교
프로젝트 내용	출생아 수 및 합계 출산율 분석을 위한 데이터 수집 2012 ~ 2020년 / 출생아 수, 합계 출산율 데이터 추출 9년간 데이터 비교를 통한 변화된 양상 확인 - 시각화
주요 업무 및 상세 역할	1) 데이터 수집 2) 데이터 추출을 위한 방법 모색 - 오류수정(축 분리, 행열 전환 등) 3) 비교를 위한 시각화 그래프
사용언어 및 개발 환경	Python, Jupyter Notebook, VSC
참고 자료	Github 링크

Main work

데이터 추출 – 비교를 위한 데이터 수정(Index)

Fertilityrate\_Baby.ipynb

C: > Users > 90x61 > projects > Fertilityrate\_Baby > Fertilityrate\_Baby.ipynb > M출생아 수 및 합계 출산율 분석 > M데이터 정보

+ 코드 + Markdown ...

# 출생아 수 및 합계 출산율 분석

## 데이터 정보

합계출산율 정의 : 여성 인당 가임기간(15~49세)에 낳을 것으로 기대되는 평균 출생아 수  
연령별 출산율(ASFR) : 1년 간 발생한 모의 연령별(15~49세) 출생아 수를 해당 연령별  
여자의 연앙인구(7월 1일 기준)로 나누어 1,000분율로 표시  
합계 출산율 (TFR) : 여성 1명이 평생동안 낳을 것으로 예상되는 평균 출생아 수를  
나타낸 지표로서 연령별 출산율(ASFR)의 총합이며, 출산력 수준을 나타내는 대표적 지표임  
기간 : 2012 ~ 2020 비교  
출처 : [https://www.index.go.kr/potal/main/EachDtlPageDetail.do?idx\\_cd=1428](https://www.index.go.kr/potal/main/EachDtlPageDetail.do?idx_cd=1428)

```
#1. 파일 불러오기
#2. 불필요한 열 건너뛰기 : 2열 제거
#3. 인덱스 설정 : 첫번째 열
#4. 해당 데이터 선택 : 불필요한 열 제거 후 첫번째는 구분으로 자동분류되고 이후 필요한 행의 개수
import pandas as pd
df = pd.read_excel('stat_142801.xls', skiprows=2, nrows=2, index_col=0)
df
```

	2012	2013	2014	2015	2016	2017	2018	2019	2020
출생아 수	484.600	436.500	435.400	438.400	406.200	357.800	326.800	302.700	272.30
합계 출산율	1.297	1.187	1.205	1.239	1.172	1.052	0.977	0.918	0.84

```
#오류 : df['출생아 수']
df.index
```

Fertilityrate\_Baby.ipynb

C: > Users > 90x61 > projects > Fertilityrate\_Baby > Fertilityrate\_Baby.ipynb > M출생아 수 및 합계 출산율 분석 > M데이터 정보

+ 코드 + Markdown ...

```
#x축(연도)을 공유하는 y축 2개(좌-출생아 수, 우-합계 출산율) 생성

#출생아 수 - 막대 그래프
fig, ax1 = plt.subplots(figsize=(13,5)) #default값 1, 1 - 1개의 그래프
fig.suptitle('출생아 수 및 합계 출산율')

ax1.set_ylabel('출생아 수 (천명)')
ax1.set_ylim(200, 700)
ax1.set_yticks([300, 400, 500, 600])
ax1.bar(df.index, df['출생아 수'], color='orange')
#데이터 표시 - 방법 1
#for idx, val in enumerate(df['출생아 수']):
#    ax1.text(idx, val + 12, val, ha='center')

#데이터 표시 - 방법 2
bar = ax1.bar(df.index, df['출생아 수'], color='orange')
for idx, rect in enumerate(bar):
    #print(rect)
    plt.text(idx, rect.get_height()+7, df['출생아 수'][idx], ha='center')

#합계 출산율 - 꺾은선 그래프
#x축 공유하는 쌍둥이 축
ax2 = ax1.twinx()
ax2.set_ylabel('합계 출산율 (가임여성 1명당 명)')
ax2.set_ylim(0, 1.5)
ax2.set_yticks([0, 0.5, 1])
line = ax2.plot(df.index, df['합계 출산율'], color='yellow', marker='o', ms=15, mec='w',
               mew=3, lw=5)
for idx, val in enumerate(df['합계 출산율']):
    ax2.text(idx, val+0.08, val, ha='center')
#오류 : for idx, rect in enumerate(line)
#ax2.text(idx, rect.get_height(), values[idx])
#bar만 가능

plt.savefig('출생아 수 및 합계 출산율 비교.png', dpi=100)
```



Project 03.

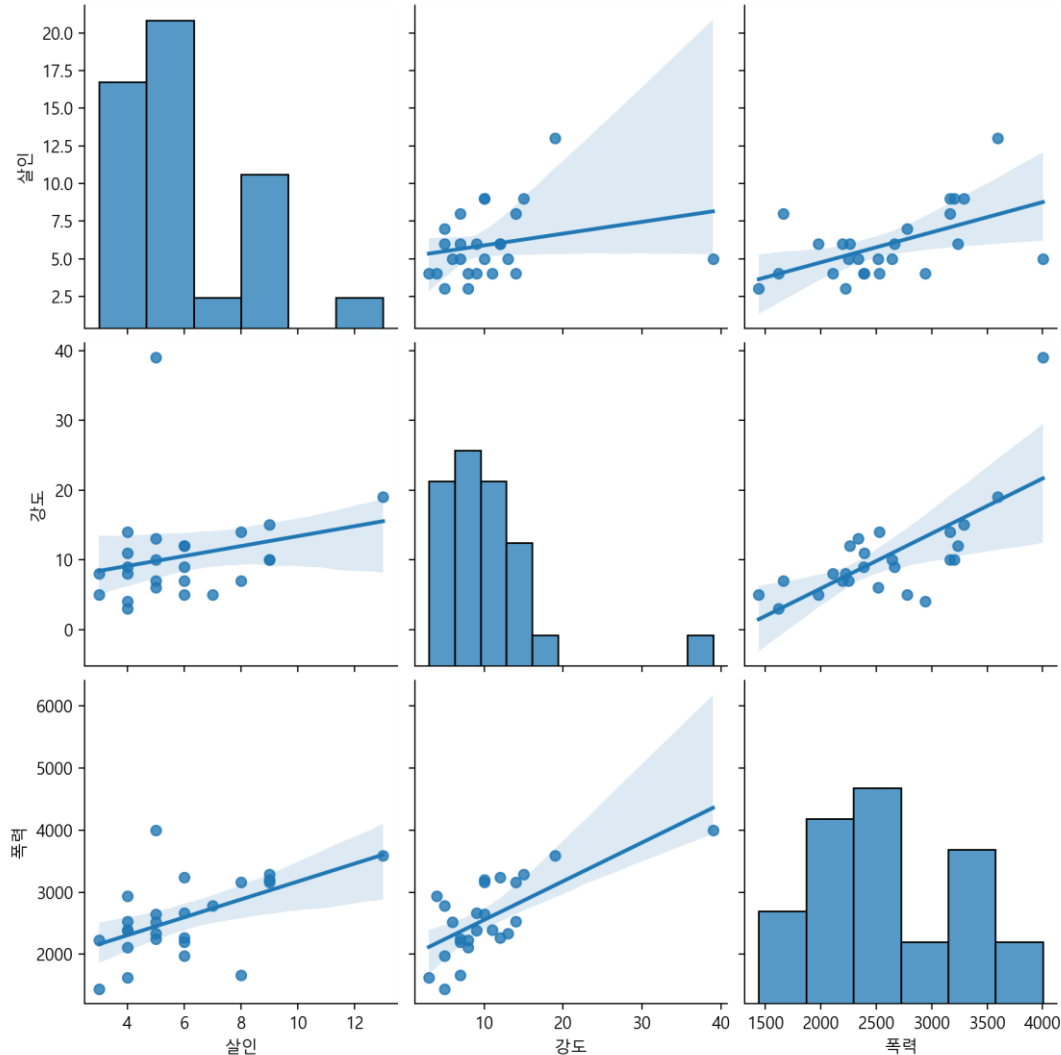
# 서울시 구별 범죄현황 분석

About project



2016년 구별 범죄 데이터를 불러오고 데이터 전처리 및 시각화  
서울시 CCTV 현황 데이터와 데이터 Join

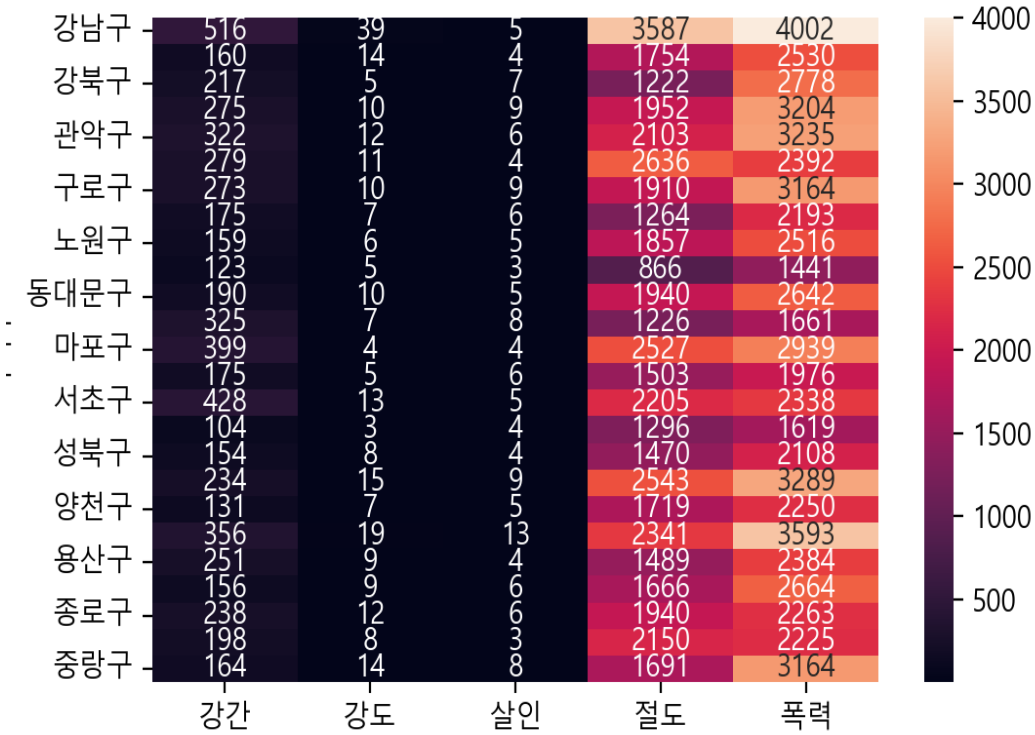
범죄(살인, 강도, 폭력) 상관관계 그래프



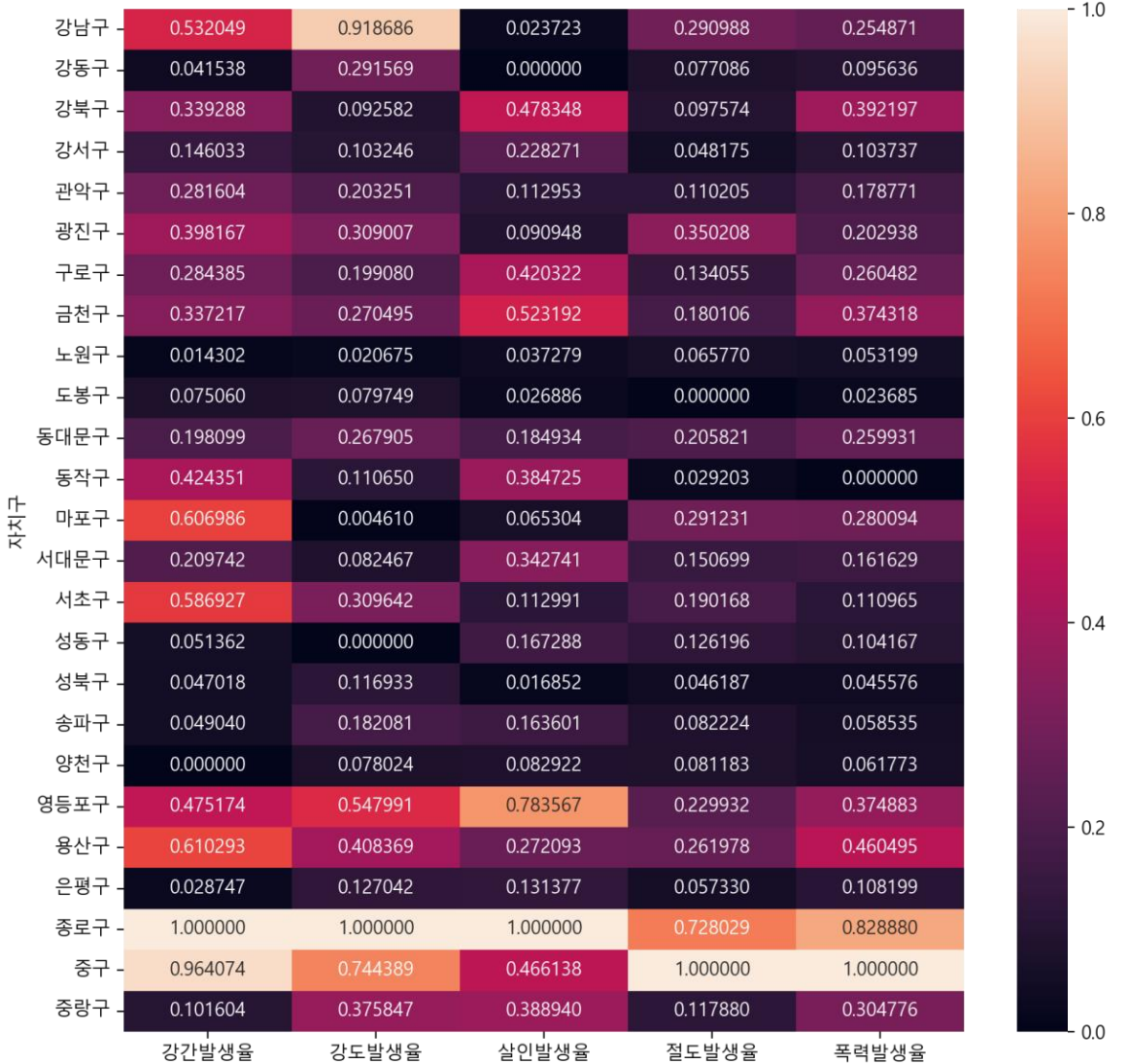
## Introduce project

작업 기간	2022. 05.
인력 구성(기여도)	BE 1명 / FE 1명 (FE 기여도 100%)
프로젝트 목적	2016년 구별 범죄 데이터를 불러오고 데이터 전처리 및 시각화 서울시 구별 CCTV 현황 데이터 병합
프로젝트 내용	서울시 구별 범죄현황 위한 데이터 수집 2016년 서울 기준 - 데이터 추출 및 범죄 수, 범죄율 도출 범죄 간 상관관계 및 자치구별 발생 현황 - 시각화
주요 업무 및 상세 역할	1) 데이터 수집 2) 데이터 추출을 위한 방법 모색 : 자치구 종합, 범죄율 등 3) 상관관계 및 정도를 위한 시각화 그래프
사용언어 및 개발 환경	Python, Jupyter Notebook, VSC
참고 자료	Github 링크
기타사항 (아쉬운 점)	결과 도출 미비 : 범죄 데이터와 CCTV간의 관계 분석 예정

자치구 별 범죄 발생 수 시각화



자치구 별 범죄 발생율 시각화



# Project 02. 서울시 구별 범죄현황 분석

## Main work 데이터 추출 – 데이터 시각화(그래프, heatmap 등)

Python\_Seoul Criminal.ipynb

C: > Users > 90x61 > projects > Python\_Seoul Criminal.ipynb > M서울시 구별 범죄현황

+ 코드 + Markdown | ▶ 모두 실행 ≡ 모든 셀의 출력 지우기 | ≡ Outline ...

### 불필요한 행과 컬럼 제거

```
#불필요한 컬럼 제거하기
criminal.drop(['기간', '합계검거', '살인검거', '강도검거', '강간검거', '절도검거', '폭력검거'], axis=1, inplace=True)
criminal.drop([0], inplace=True)

#컬럼 이름 설정
criminal.rename(columns={'살인발생': '살인', '강도발생': '강도', '강간발생': '강간', '절도발생': '절도',
                        '합계발생': '합계', '폭력발생': '폭력'}, inplace=True)
print(criminal.head(5))
```

	자치구	합계발생	살인	강도	강간	절도	폭력
1	종로구	4459	6	12	238	1940	2263
2	중구	4584	3	8	198	2150	2225
3	용산구	4137	4	9	251	1489	2384
4	성동구	3026	4	3	104	1296	1619
5	광진구	5322	4	11	279	2636	2392

### 인구수 통합

```
pop = pd.read_csv('c://programming/python/pandasdata/pop.txt', encoding='utf-8', skiprows=2, delimiter='\t',
thousands=',')
pop = pop[['자치구', '계']]
print(pop.head(5))
print()

df = pd.merge(criminal, pop)
print(df.head())
```

cctv와 df의 join - df의 자치구와 cctv의 기관명으로 조인

df = pd.merge(df, cctv, left\_on='자치구', right\_on='기관명')
df.head()

...

	자치구	합계발생	살인	강도	강간	절도	폭력	계	강간발생율	강도발생율	절도발생율	폭력발생율	살인발생율	기관명	중계
0	종로구	4459	6	12	238	1940	2263	153789	0.001548	0.000078	0.012615	0.014715	0.000039	종로구	1772
1	중구	4584	3	8	198	2150	2225	131787	0.001502	0.000061	0.016314	0.016883	0.000023	중구	2333
2	용산구	4137	4	9	251	1489	2384	237285	0.001058	0.000038	0.006275	0.010047	0.000017	용산구	2383
3	성동구	3026	4	3	104	1296	1619	292672	0.000355	0.000010	0.004428	0.005532	0.000014	성동구	3602
4	광진구	5322	4	11	279	2636	2392	352627	0.000791	0.000031	0.007475	0.006783	0.000011	광진구	2588

#인덱스 설정
df = pd.pivot\_table(df, index='자치구', aggfunc=np.sum)
print(df)

Output exceeds the size limit. Open the full output data in a text editor:

자치구	강간	강간발생율	강도	강도발생율	계	살인	살인발생율	절도	절도발생율	중계	\
강남구	516	0.000959	39	0.000073	537800	5	0.000009	3587	0.006670	6502	
강동구	160	0.000343	14	0.000030	466472	4	0.000009	1754	0.003760	2547	
강북구	217	0.000717	5	0.000017	302563	7	0.000023	1222	0.004039	2462	
강서구	275	0.000474	10	0.000017	579708	9	0.000016	1952	0.003367	2560	
관악구	322	0.000645	12	0.000024	499440	6	0.000012	2103	0.004211	4042	
금천구	270	0.000791	11	0.000031	352627	4	0.000011	2636	0.007475	2588	
구로구	273	0.000648	10	0.000024	421163	9	0.000021	1910	0.004535	4875	
금천구	175	0.000715	7	0.000029	244891	6	0.000025	1264	0.005161	2374	

데이터 표준화

#강간발생율, 강도발생율, 살인발생율, 절도발생율, 폭력발생율을 MinMaxScaling
from sklearn import preprocessing

col = ['강간발생율', '강도발생율', '살인발생율', '절도발생율', '폭력발생율']

#numpy의 배열로 추출
x = df[col].values
x

min\_max\_scaler = preprocessing.MinMaxScaler()
x\_scaled = min\_max\_scaler.fit\_transform(x.astype(float))
x\_scaled

criminal\_norm = pd.DataFrame(x\_scaled, columns=col, index=df.index)
print(criminal\_norm)

...

	자치구	강간발생율	강도발생율	살인발생율	절도발생율	폭력발생율
강남구	0.532049	0.918686	0.023723	0.290988	0.254871	
강동구	0.041538	0.291569	0.000000	0.077086	0.095636	
강북구	0.339288	0.092582	0.478348	0.097574	0.392197	
강서구	0.146033	0.103246	0.228271	0.048175	0.103737	
관악구	0.281604	0.203251	0.112953	0.110205	0.178771	
금천구	0.398167	0.309007	0.090948	0.350208	0.202938	
구로구	0.284385	0.199080	0.420322	0.134055	0.260482	
금천구	0.337217	0.270495	0.523192	0.180106	0.374318	

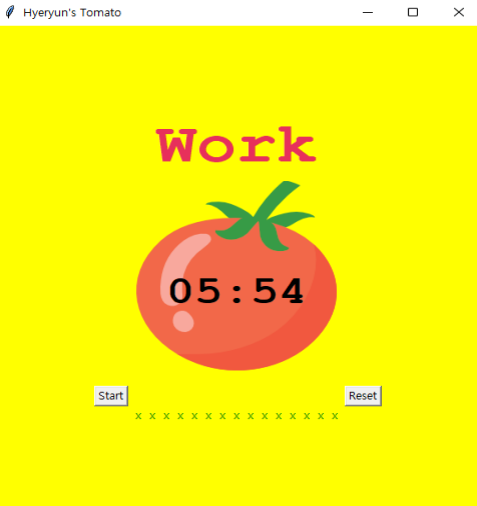
Project 04.

# 소규모 Program

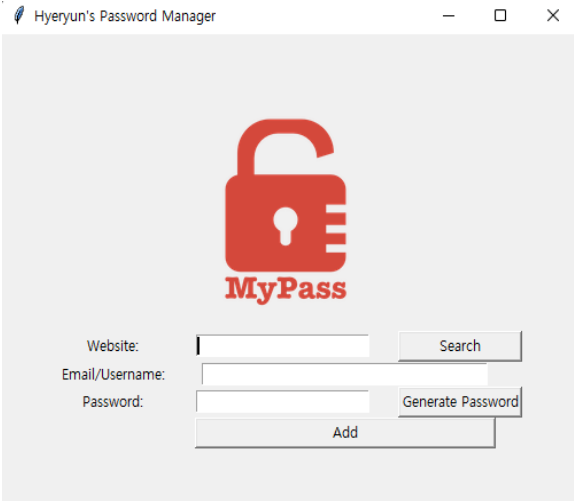
About project

Python 언어를 이용한 소규모 program 만들기

Timer – start/reset/count



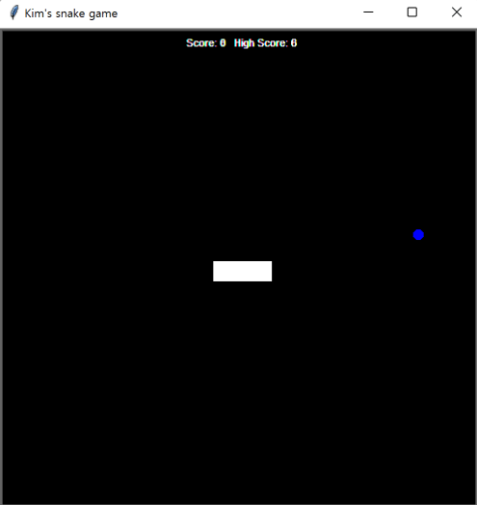
Password Manager



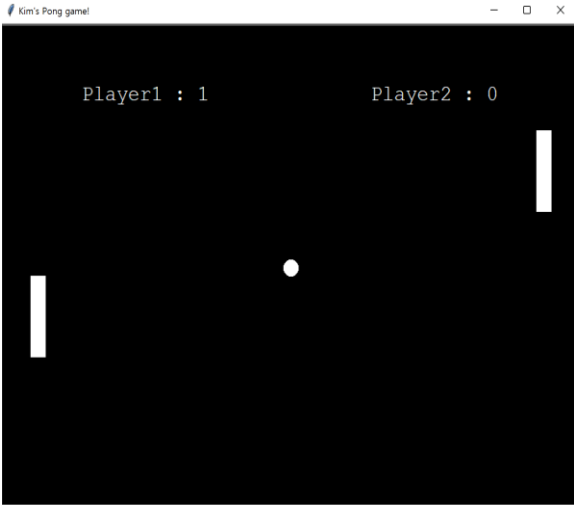
Introduce project

작업 기간	2022. 05.	
인력 구성(기여도)	BE 1명 / FE 1명 (FE 기여도 100%)	
프로젝트 목적	Python 언어를 이용한 소규모 program 만들기	
프로젝트 내용	각 program에 대한 기획 – GUI, 운영방법 등 Python을 활용한 코드 작성방법, 클래스 생성, 오류 분석 등	
주요 업무 및 상세 역할	Timer	1) Start – 5sec 카운트 다운 후 시작 2) 25min(work)-5(break) * 3회 반복 후 마지막 20min(break) 3) work에 대한 횟수 표시 – 하단 중앙(
	Password Manager	1) website, email/username, password 입력 후 add 시 저장 – 미입력 시 저장 불가(경고창) 2) website로 검색 가능 – popup 3) password 무작위 생성 – 자동 복사 기능
	Sneak Game	1) 무작위 아이템 형성 2) 아이템 먹게되면 sneak 길이 길어짐 3) 최고 스코어 반영
	Ping Pong Game	1) 2명의 탁구 게임 2) 위,아래 부딪힐 경우 튕겨서 나오며 좌.우 농칠 경우 점수 획득/차감 3) 키보드 조작법 – w,s / up,down 4) 점수 업데이트
사용언어 및 개발 환경	Python, Pycharm	
참고 자료	Github 링크	

Sneak Game



Ping Pong Game



# Project 04. 소규모 Program – Timer

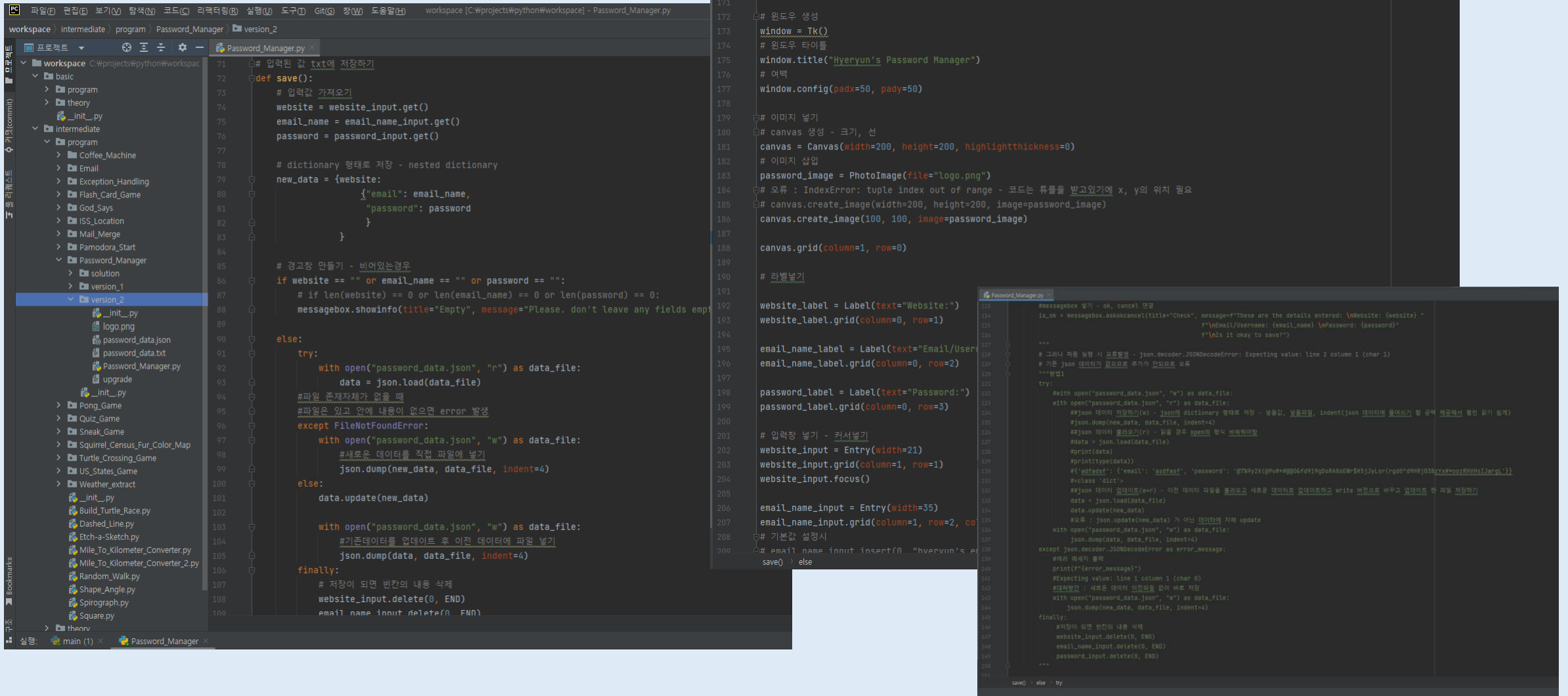
## Main work Timer mechanism – start/reset, countdown, time format, work\_count

```
workspace | Intermediate | program | Pamodora_Start | main.py
main.py
106 # ----- UI SETUP ----- #
107 from tkinter import *
108 #윈도우 생성
109 window = Tk()
110 #타이틀
111 window.title("Hyeryun's Tomato")
112 #어색상성 - 윈도우의 배경색
113 window.config(padx=100, pady=100, bg="YELLOW")
114
115
116
117 #카운트 다운
118 #오류 : NameError: name 'canvas' is not defined - canvas 정의된 이후로 생성
119 #count_down(5)
120
121
122
123 #라벨 생성
124 timer_label = Label(text="Timer", font=(FONT_NAME, 50, "bold"))
125 timer_label.config(bg="YELLOW", fg="GREEN")
126 #오류 - timer_label.config(bg=YELLOW, fg="Blue") : 변수 또한 ""
127 timer_label.grid(column=1, row=0)
128
129 #전반적인 토마토 사진에 대한 control
130 #canvas 생성 - 영역크기 설정 - 토마토의 배경색 - 토마토 이미지의 테두리 제거
131 canvas = Canvas(width=200, height=240, bg="YELLOW", highlightthickness=0)
132 #canvas 위에 얹어 쓸 것들 표현하기 - canvas.create....
133 ###이미지의 경우 tkinter 에서 이미지를 먼저 읽어와야함
134 tomato_image = PhotoImage(file="tomato.png")
135 canvas.create_image(100, 110, image=tomato_image)
136
137 #canvas를 통해 text - 0, 0은 좌상단에서 부터 시작하며 숫자가 증가할수록 오른쪽, 아래로 이동
138 time_text = canvas.create_text(100, 130, text="00:00", font=(FONT_NAME, 35, "bold"))
139
140 #canvas 늘기
141 canvas.grid(column=1, row=1)
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

```
main.py
64 #오류 : GUI 프로그램 내에서 발생 - 누르고 새로고침 되는 이벤트를 계속 주시 - 이런 GUI 프로그램 : EVENT Driven
65 #메인 루프 내에 또다른 루프 존재하는 경우로 아래 코드는 메인 루프에 도달하지 못할
66 '''
67 import time
68
69 count = 5
70 while True:
71     time.sleep(1)
72     count -= 1
73 '''
74
75 #window.after 함수는 시간 대기 후 실행
76 def count_down(count):
77     # 300 -> 05:00 시간형식으로 표시
78     #소수점 존재로 math 함수에서 floor
79     count_min = math.floor(count / 60)
80     count_sec = count % 60
81
82     #00으로 표시하기
83     if count_sec == 0: --- int
84     # count_sec = "00" --- str
85     if count_min < 10:
86         count_min = f"0{count_min}"
87     if count_sec < 10:
88         count_sec = f"0{count_sec}"
89
90     # 해당 카운트를 화면에 표시 - 직접 지정
91     canvas.itemconfig(time_text, text=f"{count_min}:{count_sec}")
92     # 오류 : AttributeError: 'int' object has no attribute 'config'
93     # time_text.config(text=f"{count}")
94
95     # 1초 후 대기 후 count, 1씩 감소
96     if count > 0:
97         global timer_continue
98         timer_continue = window.after(1, count_down, count -1)
99     #00:00에 도달할 경우
100     else:
101         start_timer()
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

```
main.py
64 #오류 : GUI 프로그램 내에서 발생 - 누르고 새로고침 되는 이벤트를 계속 주시 - 이런 GUI 프로그램 : EVENT Driven
65 #메인 루프 내에 또다른 루프 존재하는 경우로 아래 코드는 메인 루프에 도달하지 못할
66 '''
67 import time
68
69 count = 5
70 while True:
71     time.sleep(1)
72     count -= 1
73 '''
74
75 #window.after 함수는 시간 대기 후 실행
76 def count_down(count):
77     # 300 -> 05:00 시간형식으로 표시
78     #소수점 존재로 math 함수에서 floor
79     count_min = math.floor(count / 60)
80     count_sec = count % 60
81
82     #00으로 표시하기
83     if count_sec == 0: --- int
84     # count_sec = "00" --- str
85     if count_min < 10:
86         count_min = f"0{count_min}"
87     if count_sec < 10:
88         count_sec = f"0{count_sec}"
89
90     # 해당 카운트를 화면에 표시 - 직접 지정
91     canvas.itemconfig(time_text, text=f"{count_min}:{count_sec}")
92     # 오류 : AttributeError: 'int' object has no attribute 'config'
93     # time_text.config(text=f"{count}")
94
95     # 1초 후 대기 후 count, 1씩 감소
96     if count > 0:
97         global timer_continue
98         timer_continue = window.after(1, count_down, count -1)
99     #00:00에 도달할 경우
100     else:
101         start_timer()
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

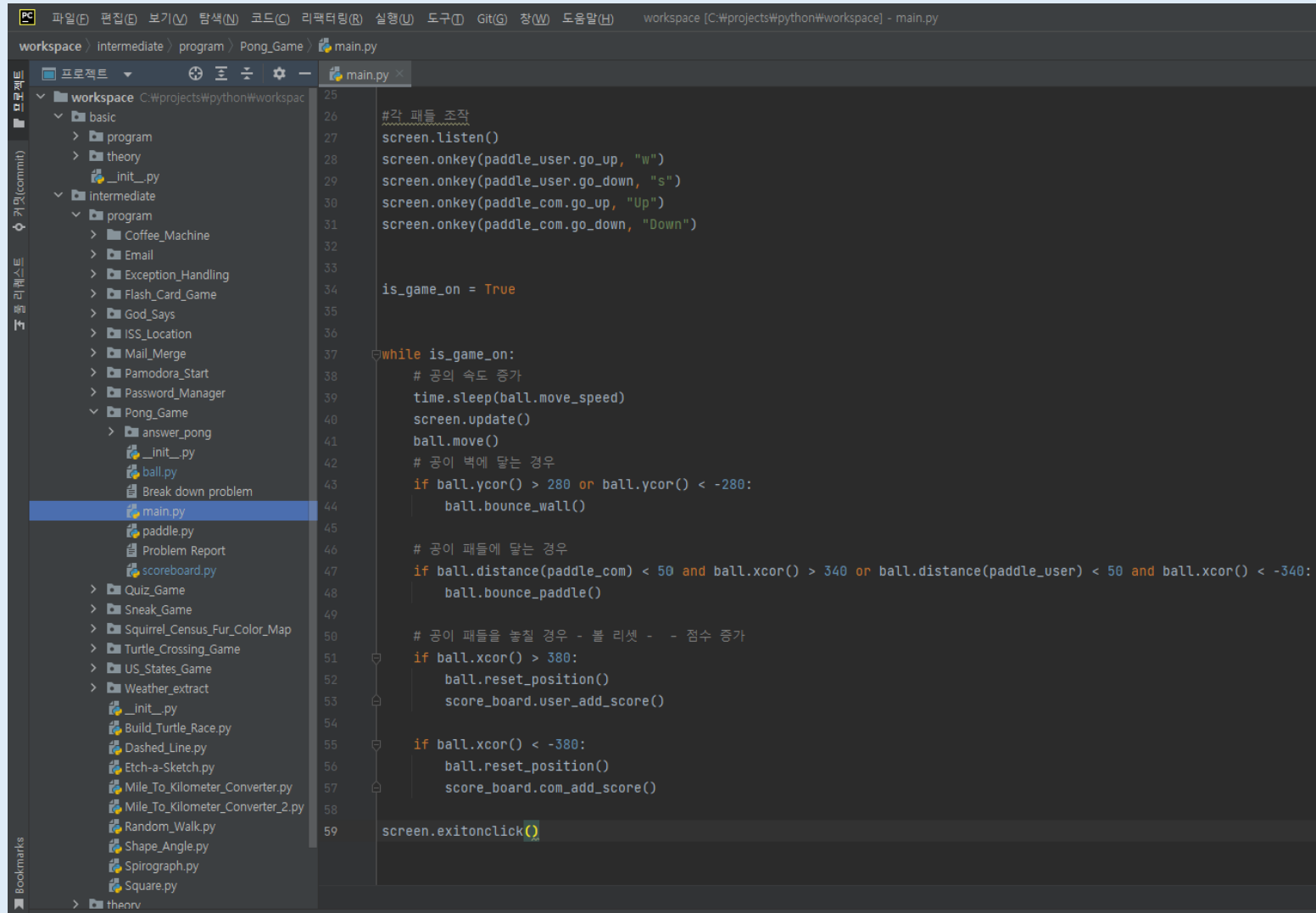
## Main work Password Generator, Data save, error message, auto\_copy



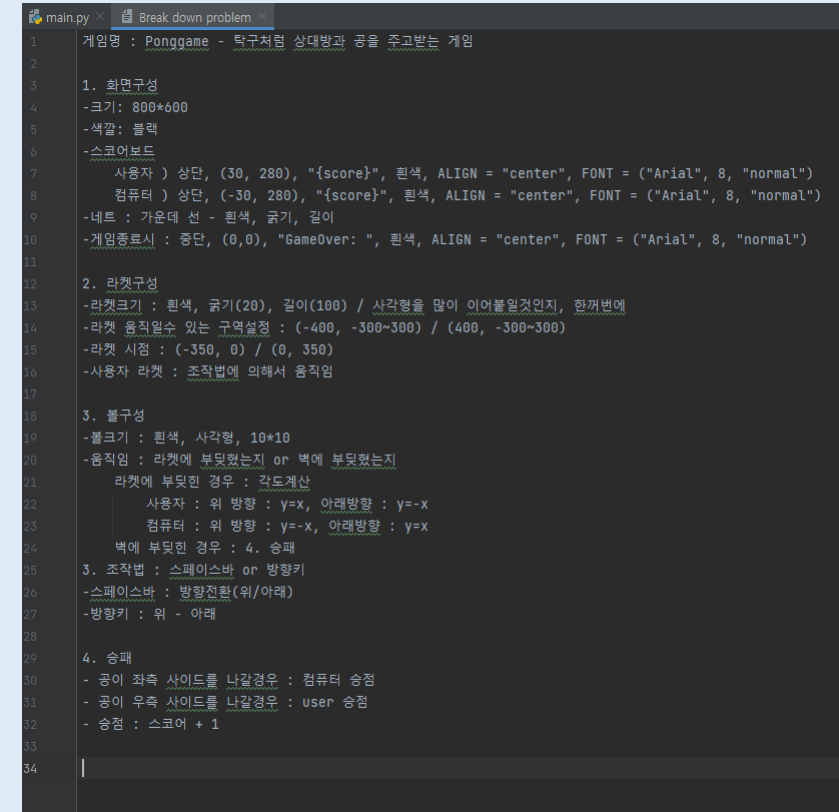
```
71 # 입력된 값 txt에 저장하기
72 def save():
73     # 입력값 가져오기
74     website = website_input.get()
75     email_name = email_name_input.get()
76     password = password_input.get()
77
78     # dictionary 형태로 저장 - nested dictionary
79     new_data = {website:
80                 {"email": email_name,
81                  "password": password
82                 }}
83
84
85
86 # 경고창 만들기 - 비어있는경우
87 if website == "" or email_name == "" or password == "":
88     # if len(website) == 0 or len(email_name) == 0 or len(password) == 0:
89     messagebox.showinfo(title="Empty", message="Please. don't leave any fields empty")
90
91 else:
92     try:
93         with open("password_data.json", "r") as data_file:
94             data = json.load(data_file)
95
96         #파일 존재자체가 없을 때
97         #파일을 읽고 안에 내용이 없으면 error 발생
98     except FileNotFoundError:
99         with open("password_data.json", "w") as data_file:
100             #새로운 데이터를 직접 파일에 넣기
101             json.dump(new_data, data_file, indent=4)
102
103     else:
104         data.update(new_data)
105
106         with open("password_data.json", "w") as data_file:
107             #기존데이터를 업데이트 후 이전 데이터에 파일 넣기
108             json.dump(data, data_file, indent=4)
109         finally:
110             # 저장이 되면 빈칸의 내용 삭제
111             website_input.delete(0, END)
112             email_name_input.delete(0, END)
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```



## Main work    탁구게임, Paddle 및 Ball 생성, 점수 관리, 게임 운영방법 등



```
workspace [C:\projects\python\workspace] - main.py
workspace > intermediate > program > Pong_Game > main.py
main.py
25
26 #각 패들 조작
27 screen.listen()
28 screen.onkey(paddle_user.go_up, "w")
29 screen.onkey(paddle_user.go_down, "s")
30 screen.onkey(paddle_com.go_up, "Up")
31 screen.onkey(paddle_com.go_down, "Down")
32
33
34 is_game_on = True
35
36
37 while is_game_on:
38     # 공의 속도 증가
39     time.sleep(ball.move_speed)
40     screen.update()
41     ball.move()
42     # 공이 벽에 닿는 경우
43     if ball.ycor() > 280 or ball.ycor() < -280:
44         ball.bounce_wall()
45
46     # 공이 패들에 닿는 경우
47     if ball.distance(paddle_com) < 50 and ball.xcor() > 340 or ball.distance(paddle_user) < 50 and ball.xcor() < -340:
48         ball.bounce_paddle()
49
50     # 공이 패들을 놓칠 경우 - 볼 리셋 - 점수 증가
51     if ball.xcor() > 380:
52         ball.reset_position()
53         score_board.user_add_score()
54
55     if ball.xcor() < -380:
56         ball.reset_position()
57         score_board.com_add_score()
58
59 screen.exitonclick()
```



```
main.py > Break down problem
1 게임명 : Ponggame - 탁구처럼 상대방과 공을 주고받는 게임
2
3 1. 화면구성
4 -크기: 800*600
5 -색깔: 블랙
6 -스코어보드
7 사용자 ) 상단, (30, 280), "{score}", 흰색, ALIGN = "center", FONT = ("Arial", 8, "normal")
8 컴퓨터 ) 상단, (-30, 280), "{score}", 흰색, ALIGN = "center", FONT = ("Arial", 8, "normal")
9 -네트 : 가운데 선 - 흰색, 굵기, 길이
10 -게임종료시 : 중단, (0,0), "GameOver: ", 흰색, ALIGN = "center", FONT = ("Arial", 8, "normal")
11
12 2. 라켓구성
13 -라켓크기 : 흰색, 굵기(20), 길이(100) / 사각형을 많이 이어붙일것인지, 한개밖에
14 -라켓 움직일수 있는 구역설정 : (-400, -300~300) / (400, -300~300)
15 -라켓 시작 : (-350, 0) / (0, 350)
16 -사용자 라켓 : 조작법에 의해서 움직임
17
18 3. 볼구성
19 -볼크기 : 흰색, 사각형, 10*10
20 -움직임 : 라켓에 부딪혔는지 or 벽에 부딪혔는지
21 라켓에 부딪힌 경우 : 각도계산
22 사용자 : 위 방향 : y=x, 아래방향 : y=-x
23 컴퓨터 : 위 방향 : y=-x, 아래방향 : y=x
24 벽에 부딪힌 경우 : 4. 승패
25
26 3. 조작법 : 스페이스바 or 방향키
27 -스페이스바 : 방향전환(위/아래)
28 -방향키 : 위 - 아래
29
30 4. 승패
31 - 공이 좌측 사이드를 나갔을경우 : 컴퓨터 승점
32 - 공이 우측 사이드를 나갔을경우 : user 승점
33 - 승점 : 스코어 + 1
34
```

# Project 04. 소규모 Program – Ping Pong Game

## Main work Sneak / Item 생성, 방향키 조작, 점수 관리(최고점 데이터 관리 등)

```
workspace > intermediate > program > Sneak_Game > sumsquare_snake_upgrade > main_sumsquare.py
main_sumsquare.py scoreboard_sumsquare.py foody_sumsquare.py total_sumsquare.py

1 from turtle import Turtle, Screen
2 #위에 Turtle이라는 클래스 미사용 : 이유는 아래 클래스에서 상속되어 이미 사용되었기 때문에
3 from snake_sumsquare import Snake_sumsquare
4 from foody_sumsquare import Food_sumsquare
5 from scoreboard_sumsquare import Scoreboard_sumsquare
6
7 import time
8
9 screen = Screen()
10 screen.setup(width=500, height=500)
11 screen.bgcolor("black")
12 screen.title("Kim's snake game")
13 #애니메이션 끄기 - 이동간의 흐름 보지않기(update 나올때까지)
14 #사각형을 할것을경우
15 #화면 바뀔때마다의 애니메이션이 보여지는 것을 off -update 될때까지
16 screen.tracer(0)
17
18
19 #객체 생성
20 #범 객체
21 snake = Snake_sumsquare()
22 #아이템 객체
23 food = Food_sumsquare()
24 #점수판 객체
25 scoreboard = Scoreboard_sumsquare()
26
27 #조작법 설정 - 오류 : key = "파스칼표기" / up(x), up(o)
28 screen.listen()
29 screen.onkey(key="Up", fun=snake.go_up)
30 screen.onkey(key="Down", fun=snake.go_down)
31 screen.onkey(key="Left", fun=snake.go_left)
32 screen.onkey(key="Right", fun=snake.go_right)
33
34 #게임진행여부
35 is_game_on = True
36 while is_game_on:
37     움직이고 나서 update하면 갱신된 화면만 보임
38     screen.update()
39     # 1초씩 지연 : 숫자가 작아질수록 빨라짐
40     time.sleep(0.1)
41
42 # 1초씩 지연 : 숫자가 작아질수록 빨라짐
43 time.sleep(0.1)
44
45 snake.movesnake()
46
47 #아이템을 먹을경우(turtle.distance(비교대상))
48 if snake.head.distance(food) < 15: #아이템의 크기고려하여 그 간의 거리 지정
49     print("nom nom nom") #콘솔에 표시
50     #아이템 사라지고 재생성
51     food.refresh()
52     #점수 증가
53     scoreboard.increase_score()
54     #범 그리 스퀘어 하나 증가
55     snake.extendsegment()
56
57 #벽에 부딪힐 경우 - 경계선 x축 y축 -230 ~ 230
58 if snake.head.xcor() > 230 or snake.head.xcor() < -230 or snake.head.ycor() < -230 or snake.head.ycor() > 230:
59     #is_game_on = False
60     #scoreboard.game_over()
61     scoreboard.reset()
62     snake.reset()
63
64 #범의 머리가 자신의 몸통과 부딪힐 경우 - 머리와 각 몸통의 거리로 판단 (단, 자신의 몸통이 머리일 경우 제외)
65 # for segment in snake.segments:
66 #     if segment == snake.head:
67 #         pass
68 #     elif snake.head.distance(segment) < 10:
69 #         is_game_on = False
70 #         scoreboard.game_over()
71
72 #슬라이싱 사용하기
73 for segment in snake.segments[1:]:
74     if snake.head.distance(segment) < 10:
75         is_game_on = False
76         #scoreboard.game_over()
77         #점수 최소화 - 현재점수는 초기화, 최고점수는 업데이트
78         scoreboard.reset()
79         #범 초기화 - 처음자리로 오기
80         snake.reset()
```

**Continue to study...**

---

**End of Document**

---