|  |
| --- |
| **SSAC**  **파이썬을 활용한 비즈니스 빅데이터 분석가 양성 과정**  **(TEMPLATE) 자가학습 워크시트** |

|  |  |
| --- | --- |
| **이름** | 백승재 |
| **날짜** | 2021-09-20 |
| **온라인 콘텐츠(인강)**  **학습 범위** | * 직장인을 위한 Python : 0401~05-01 * Python : Numpy 05\_03 ~ 05\_05 06\_01 06\_02 |
| **복습 노트 작성 (자유 양식)** | 04-01 사칙연산 (+,-,/,\*)  % 몫을 나눈 나머지, // 나눗셈에 대한 몫, \*\* 제곱 연산  04-02 괄호 사용: 사칙연산 우선 순위 부여 시 () 사용  04-03 문자열의 연산 : + 만 지원함  05-01 주석과 Indent :  - 주석 #, “”””””  - indent: 콜론 다음에 반드시 indent를 부여할 것, 계위(Level)을 결정함  05-03. DataFrame Method  - 데이터의 상위 5개 출력 df.Head(), 하위 5개 출력 df.tail()  - Index,columns,values 출력 df.index, df.columns, df.values  - dataframe 정보 확인 : df.info()  - Aggregation 통계 정보 확인 : df.describe()  -column별 내림차순,상위 몇 개 데이터 지정 정렬:  **df.sort\_values(‘column name’, ascending=’False’).head()**  **05-04. DataFrame indexing**  - indexing : 데이터에서 어떤 특정 조건을 만족하는 원소를 찾는 방법.  - python dictionary & numpy indexing(fancy indexing)  -> 전체 DataFrame에서 조건에 만족하는 데이터를 쉽게 찾아서 조작할 때 유용하게사용  - column 단위 indexing(series) : **df[“A”]** (dict indexing과 같음, key를 indexing = “colomn”)  - row 단위 indexing(series):  1.row값 location: df.loc[‘row ‘]  2.row index location : df.iloc[3]  - slicing  1. index(정수)을 이용한 slicing : **df[index :index]**  2. index value을 이용한 slicing : **df[“index value: “index value”**  **3. 2차원 indexing, columns(key값)은 리스트로 넘겨줄수 있다.**  **row기준 > column 별로 slicing : df[row slicing, [‘column명’]]**  **4**. iloc를 사용하면 numpy array의 2차원 indexing과 같음  5. fancy indexing : key column df.A or df[“A”]  - 조건문을 통해 indexing을 할 수 있는 방법 (True와 False를 원소로 하는 리스트 이용)  **- Chain indexing: indexing이 앞에서부터 뒤로 쭉 순서대로 적용됨**  **Df[‘A’][df[“A”]>0]**  **- dataframe copy 하기 : df.copy()**  **- Column 추가/수정하기 : dict 방법과 동일 df[column] = [row value]**  **- isin : raw value argument있는지 확인 -> indexing으로 활용 가능**  06\_01\_Seaborn이란?  - Matplot library기본으로 확장되어 만들어진 library  - 그래프가 미적이다.  - 쉽게 그래프를 그릴 수 있고 데이터를 설명할 수 있다.  - 동일한 데이터를 가지고 그래프를 실습  - pair plot(scatter plot, heatmat. Plot graph  - matplot library 객체 지향적으로 어떻게 구현하여 커스터 마이징하기  06\_02\_ Histplot & Displot |
| **퀴즈 제출** | Q1. 다음 중 출력 결과가 다른 한 개는?  1) print(“Life””is””Short”)  2) print(“Life”+”is”+”Short”)  3) print(“Life”,”is”,”Short”)  4) print(“”.join([“Life””is””Short”])  정답 : 3번 ‘Life is Short’ (,는 공백을 부여함) 1,2,4 :’Lifeisshort’  Q2. 다음 dataframe을 B기준으로 하위 3개 내림차순으로 정리를 한 것으로 적절한 것은?  date\_new = pd.date\_range('20210915',periods=8)  df = pd.DataFrame(np.random.rand(8,5), index=date\_new, columns= ["A","B","C","D","E"])  df    1. df.sort\_values(“B”,ascending=”True”).head()  2. df.sort\_values(“3”,ascending=”True”).tail(B)  3. df.sort\_values(“B”,ascending=”False”).tail(3)  4. df.sort\_values(“3”,ascending=”True”).head(B)  정답 : 3  그룹별 지정은 sort\_values를 사용하여 ( ) column에 이름을 str으로 적어준다. 정렬의 기본 옵션은 오름차순이다. SQL의 limit과 같이 가장 우위 또는 하위를 조회할때는 head, tail을 사용한다.    Q3 .다음중 df dataframe을 slicing한 값 중 다른 하나는?    1. df.loc['2021-09-17':'2021-09-19',["C","D"]]  2. df.iloc[1:4,2:4]  3. df.iloc[[1,2,3],[2,3]]  4. df.iloc[1:3,2:3]  정답 : 4  4.(출력값) 1~3(출력값) |
| **.** |  |
| **화면 캡쳐**  **(교육 화면 + 교육일시)** |  |