인터페이스 개발

수행평가

멀티캠퍼스

이원용

**수행평가1**

|  |
| --- |
| **문제 1** |
| 10보다 작은 자연수 중에서 3 또는 5의 배수는 3,5,6,9가 존재해요!  이것들의 합은 23입니다.  1000보다 작은 자연수 중에서 3 또는 5의 배수들을  구해서 모두 합하면 얼마인가요?  정답 : 233168 |
| **코드** |
| three = [thr for thr in range(1,1000) if thr%3==0]  five=[fiv for fiv in range(1,1000) if fiv%5==0]  sum(set(three)|set(five)) |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 2** |
| 피보나치 수열의 각 항은 바로 앞의 항 두 개를 더한 것이 됩니다.  1과 2로 시작하는 경우 이 수열은 아래와 같습니다.  1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...  짝수이면서 4백만 이하인 모든 항을 더하면 얼마가 됩니까?  정답 : 4613732 |
| **코드** |
| def fiv(n):  if n <=1 :  return n  elif n==2 :  return n  else :  return fiv(n-1)+fiv(n-2)  sum\_f=0  for i in range(10000):  while fiv(i)<=4000000:  if fiv(i)%2==0:  sum\_f+=fiv(i) |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 3** |
| **알파벳 대소문자로 된 문자열이 주어지면, 이 문자열에서 가장 많이 사용된 알파벳이 무엇인지 출력하는 프로그램을 작성하시오.**  **단, 대소문자는 구별하지 않아요. 만약 동률이 존재하는 경우 알파벳 순으로 제일 앞에 있는 문자를 출력하세요.**  **문자열) "This is a sample Program mississippi river"**  **문자열) "abcdabcdababccddcd"**  **정답 : "This is a sample Program mississippi river" => I**  **정답 : "abcdabcdababccddcd" => C** |
| **코드** |
| def most\_alphabet(string):  d = {}  for i in string.lower():  d[i]=d.get(i,0)+1  return sorted(d.items(),key=lambda x : x[1],reverse=True)[0][0]  most\_alphabet(b) |
| **결과 이미지** |
|  |
| **문제 4** |
| 로또 프로그램 작성5000원으로 로또복권을 5장 자동으로 구매합니다.이번 주 로또 당첨번호를 생성하여 로또 당첨을 확인하세요!쉬운버전으로 먼저 작성합니다.6숫자가 다 맞으면 1등, 5개 맞으면 2등으로 처리합니다.즉, 쉬운버전은 보너스 숫자는 없는 것으로 간주합니다. |
| **코드** |
| import random  luck = []  for i in range(0,6):  luck.append(random.randint(1,100))  def lotto(n):  out=[]  for i in range(n):  luck = []  for j in range(6):  luck.append(random.randint(1,100))  out.append(luck)  return out  def check\_lotto(my\_lotto):  for i in range(len(my\_lotto)):  cnt=0  for j in range(6):  if my\_lotto[i][j] in luck:  cnt +=1  if cnt == 6 :  print("1등")  elif cnt ==5:  print('2등')  elif cnt ==4:  print('3등')  elif cnt ==3:  print('4등')  else:  print("꽝") |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 5** |
| 어떤 수를 소수의 곱으로만 나타내는 것을 소인수분해라 하고,이 소수들을 그 수의 소인수라고 합니다.예를 들면 13195의 소인수는 5, 7, 13, 29 입니다.[¶](http://localhost:8888/notebooks/K-Digital%20%ED%94%84%EB%A1%9C%EC%A0%9D%ED%8A%B8%ED%98%95%20AI%20%EC%84%9C%EB%B9%84%EC%8A%A4%20%EA%B0%9C%EB%B0%9C/TIL/%EC%88%98%ED%96%89%ED%8F%89%EA%B0%801%20%20-%20Python%20%ED%94%84%EB%A1%9C%EA%B7%B8%EB%9E%98%EB%B0%8D%20%EA%B8%B0%EB%B3%B8.ipynb#%EC%98%88%EB%A5%BC-%EB%93%A4%EB%A9%B4-13195%EC%9D%98-%EC%86%8C%EC%9D%B8%EC%88%98%EB%8A%94-5,-7,-13,-29-%EC%9E%85%EB%8B%88%EB%8B%A4.)600851475143의 소인수 중에서 가장 큰 수를 구하세요.정답 : 6857 |
| **코드** |
| import math  n=int(input('Input an integer that want to find prime factor: '))  prime=[]  factor=[]  def check\_prime(number):  if number !=1:  for factor in range(2,number):  if number%factor==0:  return False  else:  return False  return True  for i in range(1,n):  if check\_prime(i):  prime.append(i)    for i in prime:  if n%i ==0:  factor.append(i)  print(factor) |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 6** |
| 앞에서부터 읽을 때나 뒤에서부터 읽을 때나 모양이같은 수를 대칭수(palindrome)라고 부릅니다.두 자리 수를 곱해 만들 수 있는 대칭수 중가장 큰 수는 9009 (= 91 × 99) 입니다.세 자리 수를 곱해 만들 수 있는 가장 큰 대칭수를 구하세요정답 : 906609[¶](http://localhost:8888/notebooks/K-Digital%20%ED%94%84%EB%A1%9C%EC%A0%9D%ED%8A%B8%ED%98%95%20AI%20%EC%84%9C%EB%B9%84%EC%8A%A4%20%EA%B0%9C%EB%B0%9C/TIL/%EC%88%98%ED%96%89%ED%8F%89%EA%B0%801%20%20-%20Python%20%ED%94%84%EB%A1%9C%EA%B7%B8%EB%9E%98%EB%B0%8D%20%EA%B8%B0%EB%B3%B8.ipynb#%EC%A0%95%EB%8B%B5-:-906609) |
| **코드** |
| pal =[]  for i in range(100,1000):  for j in range(100,1000):  m = str(i\*j)  if m[:3]==m[:2:-1] :  pal.append(m)    max(pal) |
| **결과 이미지** |
|  |
| **문제 7** |
| 1 ~ 10 사이의 어떤 수로도 나누어 떨어지는 가장 작은 수는 2520입니다.그러면 1 ~ 20 사이의 어떤 수로도 나누어 떨어지는 가장 작은 수는 얼마입니까?정답 : 232792560 |
| **코드** |
| def prime\_checker(number):  if number !=1:  for factor in range(2,number):  if number%factor==0:  return False  else:  return False  return True  primeunder=[]  def prime\_gen(n):  for i in range(2,n+1):  if prime\_checker(i)==True:  primeunder.append(i)      def product(arr):  ans=1  for i in arr:  ans = ans\*i  return ans    n=int(input('몇 까지의 최소공배수를 구할까요?:'))  prime\_gen(n)  p=product(primeunder)  count=0  k=1  while count<=n:  for i in range(1,n+1):  if (k\*p)%i == 0 :  count=count+1  else:  k=k+1  count=0    print(k\*p) |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 8** |
| 1부터 10까지 자연수를 각각 제곱해 더하면 다음과 같습니다 (제곱의 합).1\*\*2 + 2\*\*2 + ... + 10\*\*2 = 3851부터 10을 먼저 더한 다음에 그 결과를 제곱하면 다음과 같습니다 (합의 제곱).(1 + 2 + ... + 10)\*\*2 = 552 = 3025따라서 1부터 10까지 자연수에 대해 "합의 제곱"과 "제곱의 합" 의차이는 3025 - 385 = 2640 이 됩니다.그러면 1부터 100까지 자연수에 대해 "합의 제곱"과 "제곱의 합"의 차이는얼마입니까?정답 : 25164150[¶](http://localhost:8888/notebooks/K-Digital%20%ED%94%84%EB%A1%9C%EC%A0%9D%ED%8A%B8%ED%98%95%20AI%20%EC%84%9C%EB%B9%84%EC%8A%A4%20%EA%B0%9C%EB%B0%9C/TIL/%EC%88%98%ED%96%89%ED%8F%89%EA%B0%801%20%20-%20Python%20%ED%94%84%EB%A1%9C%EA%B7%B8%EB%9E%98%EB%B0%8D%20%EA%B8%B0%EB%B3%B8.ipynb#%EC%A0%95%EB%8B%B5-:-25164150) |
| **코드** |
| def sqsum(n):  ssum=0  for i in range(1,n+1):  ssum+=i\*\*2  return ssum  def sumsq(n):  ssum=0  for i in range(1,n+1):  ssum +=i  return ssum\*\*2  sumsq(100)-sqsum(100) |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 9** |
| 소수를 크기 순으로 나열하면 2, 3, 5, 7, 11, 13, ... 과 같이 됩니다.이 때 10,001번째의 소수를 구하세요.정답 : 104743 |
| **코드** |
| def prime(n):  primes = [2]  a = 3  while len(primes) <= n:  func = True  for b in primes:  if a % b == 0:  func = False  break  if func == True:  primes.append(a)  a = a + 2  return primes  print(prime(10001)[10000]) |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 10** |
| 세 자연수 a, b, c 가 피타고라스 정리 a****2 + b****2 = c\*\*2 를 만족하면피타고라스 수라고 부릅니다 (여기서 a < b < c ).예를 들면 3\*\*2 + 4\*\*2 = 9 + 16 = 25 = 5\*\*2이므로3, 4, 5는 피타고라스 수입니다.a + b + c = 1000 인 피타고라스 수 a, b, c는 한 가지 뿐입니다.이 때, a × b × c 는 얼마입니까?정답 : 31875000 |
| **코드** |
| for a in range(1,1000) :  for b in range(1,1000):  c =1000-a-b  if a\*a+b\*b==c\*c:  print(a\*b\*c) |
| **결과 이미지** |
|  |

**수행평가2**

|  |
| --- |
| **전처리 코드** |
| **import pandas as pd**  **my\_file = pd.read\_csv('mpg.txt')**  **class Car(object):**  **def \_\_init\_\_(self,f):**  **self.manufacturer = f['manufacturer']**  **self.model = f['model']**  **self.displ = f['displ']**  **self.year = f['year']**  **self.cyl = f['cyl']**  **self.trans = f['trans']**  **self.drv = f['drv']**  **self.cty = f['cty']**  **self.hwy = f['hwy']**  **self.fl = f['fl']**  **self.carclass = f['class']**  **self.avg = (f['cty'] + f['hwy'])/2**  **car = Car(my\_file)** |

|  |
| --- |
| **문제 1** |
| displ(배기량)이 4 이하인 자동차와 5 이상인 자동차 중 어떤 자동차의 hwy(고속도로 연비)가 평균적으로 더 높은지 확인하세요. |
| **코드** |
| car.hwy[car.displ<4].mean()  car.hwy[car.displ>5].mean() |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 2** |
| 자동차 제조 회사에 따라 도시 연비가 다른지 알아보려고 한다.  "audi"와 "toyota" 중 어느 manufacturer(제조회사)의 cty(도시 연비)가  평균적으로 더 높은지 확인하세요. |
| **코드** |
| car.cty[car.manufacturer=='audi'].mean()  car.cty[car.manufacturer=='toyota'].mean() |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 3** |
| "chevrolet", "ford", "honda" 자동차의 고속도로 연비 평균을 알아보려고 한다.  이 회사들의 데이터를 추출한 후 hwy(고속도로 연비) 평균을 구하세요.. |
| **코드** |
| car.hwy[car.manufacturer =='chevrolet'].mean()  car.hwy[car.manufacturer =='ford'].mean()  car.hwy[car.manufacturer =='honda'].mean() |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 4** |
| "audi"에서 생산한 자동차 중에 어떤 자동차 모델의 hwy(고속도로 연비)가 높은지 알아보려고 한다.  “audi"에서 생산한 자동차 중 hwy가 1~5위에 해당하는 자동차의 데이터를 출력하세요. |
| **코드** |
| car.model[(car.manufacturer=='audi')&(car.hwy.sort\_values(ascending=False))][:5] |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 5** |
| mpg 데이터는 연비를 나타내는 변수가 2개입니다.  두 변수를 각각 활용하는 대신 하나의 통합 연비 변수를 만들어 사용하려 합니다.  평균 연비 변수는 두 연비(고속도로와 도시)의 평균을 이용합니다.  회사별로 "suv" 자동차의 평균 연비를 구한후 내림차순으로 정렬한 후  1~5위까지 데이터를 출력하세요. |
| **코드** |
| car.avg[car.carclass=='suv'].sort\_values(ascending=False)[:5] |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 6** |
| mpg 데이터의 class는 "suv", "compact" 등 자동차의 특징에 따라 일곱 종류로 분류한 변수입니다.  어떤 차종의 도시 연비가 높은지 비교하려 합니다.  class별 cty 평균을 구하고 cty 평균이 높은 순으로 정렬해 출력하세요. |
| **코드** |
| for a in car.carclass.unique():  print(a,car.cty[car.carclass==a].mean()) |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 7** |
| 어떤 회사 자동차의 hwy(고속도로 연비)가 가장 높은지 알아보려 합니다.  hwy(고속도로 연비) 평균이 가장 높은 회사 세 곳을 출력하세요. |
| **코드** |
| car.hwy[car.displ<4].mean()  car.hwy[car.displ>5].mean() |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 8** |
| 어떤 회사에서 "compact" 차종을 가장 많이 생산하는지 알아보려고 합니다.  각 회사별 "compact" 차종 수를 내림차순으로 정렬해 출력하세요. |
| **코드** |
| manu\_cnt = {}  for i in car.manufacturer[car.carclass=='compact'] :  manu\_cnt[i] = manu\_cnt.get(i, 0) +1  sorted(manu\_cnt.items(),key=lambda x : x[1],reverse=True) |
| **결과 이미지** |
|  |

**수행평가3**

|  |
| --- |
| **문제 1** |
| 사용자가 평가한 모든 영화의 전체 평균 평점을 출력하세요. |
| **코드** |
| ratings['rating'].mean() |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 2** |
| 각 사용자별 평균 평점을 구하세요. 출력시 정렬은 userId로 오름차순 정렬합니다 |
| **코드** |
| ratings.groupby('userId')['rating'].mean() |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 3** |
| 각 영화별 평균 평점을 구하세요. 출력시 정렬은 movieId로 오름차순 정렬합니다. |
| **코드** |
| pd.merge(movies,ratings,on='movieId',  how='left',).groupby('movieId').mean() |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 4** |
| 평균 평점이 가장 높은 영화의 제목을 출력하세요. 단, 동률이 있을 경우 모두 출력하고 title을 기준으로 오름차순 정렬하세요. |
| **코드** |
| mr = pd.merge(movies,ratings,on='movieId',  how='left',).groupby('movieId').mean() |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 5** |
| Comedy영화 중 가장 평점이 낮은 영화의 제목을 출력하세요. 단, 동률이 있을 경우 모두 출력하고 title을 기준으로 오름차순 정렬하세요. |
| **코드** |
| mr = pd.merge(movies,ratings,on='movieId',how='outer')  mr[(mr['genres']=='Comedy')&(mr['rating']==mr['rating'].min())] |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 6** |
| 2015년도에 평가된 모든 Romance 영화의 평균 평점은? |
| **코드** |
| import datetime  now = '1442154124'  \_date = datetime.datetime.fromtimestamp(int(now)).strftime('%Y') |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 2-1** |
| displ(배기량)이 4 이하인 자동차와 5 이상인 자동차 중 어떤 자동차의 hwy(고속도로 연비)가 평균적으로 더 높은지 확인하세요. |
| **코드** |
| mpg[(mpg['displ']<4)]['hwy'].mean()  mpg[(mpg['displ']>5)]['hwy'].mean() |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 2** |
| 자동차 제조 회사에 따라 도시 연비가 다른지 알아보려고 한다.  "audi"와 "toyota" 중 어느 manufacturer(제조회사)의 cty(도시 연비)가  평균적으로 더 높은지 확인하세요. |
| **코드** |
| car.cty[car.manufacturer=='audi'].mean()  car.cty[car.manufacturer=='toyota'].mean() |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 3** |
| "chevrolet", "ford", "honda" 자동차의 고속도로 연비 평균을 알아보려고 한다.  이 회사들의 데이터를 추출한 후 hwy(고속도로 연비) 평균을 구하세요.. |
| **코드** |
| mpg[mpg['manufacturer']=='chevrolet']['hwy'].mean()  mpg[mpg['manufacturer']=='ford']['hwy'].mean()  mpg[mpg['manufacturer']=='honda']['hwy'].mean() |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 4** |
| "audi"에서 생산한 자동차 중에 어떤 자동차 모델의 hwy(고속도로 연비)가 높은지 알아보려고 한다.  “audi"에서 생산한 자동차 중 hwy가 1~5위에 해당하는 자동차의 데이터를 출력하세요. |
| **코드** |
| mpg[mpg['manufacturer']=='audi'][['hwy','model']].sort\_values(by='hwy', ascending=False)[:5] |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 5** |
| mpg 데이터는 연비를 나타내는 변수가 2개입니다.  두 변수를 각각 활용하는 대신 하나의 통합 연비 변수를 만들어 사용하려 합니다.  평균 연비 변수는 두 연비(고속도로와 도시)의 평균을 이용합니다.  회사별로 "suv" 자동차의 평균 연비를 구한후 내림차순으로 정렬한 후  1~5위까지 데이터를 출력하세요. |
| **코드** |
| mpg['avg\_y'] = (mpg['cty']+mpg['hwy'])/2  mpg[mpg['class']=='suv'].sort\_values(by='avg\_y',ascending=False)[:5] |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 6** |
| mpg 데이터의 class는 "suv", "compact" 등 자동차의 특징에 따라 일곱 종류로 분류한 변수입니다.  어떤 차종의 도시 연비가 높은지 비교하려 합니다.  class별 cty 평균을 구하고 cty 평균이 높은 순으로 정렬해 출력하세요. |
| **코드** |
| mpg.groupby(by = 'class',sort=False)['cty'].mean().sort\_values(ascending=False) |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 7** |
| 어떤 회사 자동차의 hwy(고속도로 연비)가 가장 높은지 알아보려 합니다.  hwy(고속도로 연비) 평균이 가장 높은 회사 세 곳을 출력하세요. |
| **코드** |
| mpg.groupby(by = 'manufacturer',sort=False)['hwy'].mean().sort\_values(ascending=False)[:3] |
| **결과 이미지** |
|  |

|  |
| --- |
| **문제 8** |
| 어떤 회사에서 "compact" 차종을 가장 많이 생산하는지 알아보려고 합니다.  각 회사별 "compact" 차종 수를 내림차순으로 정렬해 출력하세요. |
| **코드** |
| manu\_cnt = {}  for i in mpg[mpg['class']=='compact']['manufacturer'] :  manu\_cnt[i] = manu\_cnt.get(i, 0) +1  print(manu\_춧) |
| **결과 이미지** |
|  |