HANKUK UNIVERSITY OF FOREIGN STUDIES

과제번호: 프로그래밍과제 1
이름: 정재윤
학번: 201703262
학과: Global Business & Technology 학과
제출일자: 2021.03.16

1번 문항 문제기술

문제1)

2보다 큰 모든 짝수는 두 개의 소수(prime number)의 합으로 나타낼 수 있다는 Goldbach의 추측이 있다. 예를 들어 10 = 3 + 7, 20 = 7 + 13으로 나타낼 수 있다. 2보다 큰 임의의 짝수를 입력하여 이 짝수를 두 소수의 합으로 표현할 수 있는지를 결정하고 만약 표현할 수 있으면 차이가 가장 작은 두 소수의 합으로 나타내시오. 예를 들어 10 = 3 + 7 = 5 + 5이다. 문제에서 요구하는 결과는 5 + 5이다.

단, n이 소수인지 판별하는 함수를 정의하고 이를 사용하여야 함

입력 예

10 // 2000이하 정수

출력 예

5 5

알고리즘 및 자료구조 설명

느낀 점

프로그램 코드

```
def solution(n):
    num=[]#값의 집합
    for i in range(n//2,n+1): #1~n 까지의 수들로 원소 형성(0 은 연산해도 소수가 아니기때문)
        for j in range(1,n//2+1):
            if i>=j: #두 수가 중복되지 않도록 크기를 지정
                if n==i+j:
                    num.append([i,j]) # 합이 n 이 되는 두 수를 저장
    print(num) #테스트용
    number=[] #중복을 제거(합이 n 이 되는 두 수를 저장)
    for i in range(len(num)):
        for j in range(len(num)):
            if [num[i][0],num[i][1]]!=number:
                number.append([num[i][0],num[i][1]])

    #n 까지의 소수 판별
    pri=[]
    for A in range(2, n+1): #2 부터 1000 까지 범위의 수
        for i in range(2, n+1): #n 까지의 수 중 i 로 지정된 숫자마다의 값을 대입
            if A % i ==0: #만약 A / i 했을 때 0 의 값이 나온다면 소수임이 맞으므로 if 의 순환을 멈춘다.
                pri.append(i)
                break

    pri = list(set(pri))
    print(pri) #테스트용
    result =[]#결과 값을 담을 list

    for i in range(len(num)): #합이 n 이 되는 두 수 list
        for j in range(len(pri)): #n 까지의 소수 list
            for k in range(len(pri)):
                if num[i][0]==pri[j] and num[i][1]==pri[k]:
                    #if num[i][0]<=num[i][1]: #중복 값을 제거하고 오름차순으로 두 수를 정렬
```

```
                    print(num[i][0] , num[i][1])
                    result.append(num[i])
    print(result)

    answer=[] #최종답
    for i in range(len(result)):
        for j in range(len(result)):
            while result[i][1]-result[i][0] >= result[j][1]-result[j][0]:
                break
        break
    answer.append(result[i])
    return answer

if __name__=='__main__':
    n=int(input('정수를 입력하시오: '))
    sol=solution(n)
    print(sol)
```

2-1번 문항 문제기술

문제2)

2-1) 4지 선다형 문제들의 정답과 문제별 배점이 주어질 때, 제출한 답안지의 점수를 계산하는 프로그램을 작성하시오.

입력 예:

2 4 1 3 3 2 # 문제별 정답 (1번 문제 정답: 2, 2번 문제 정답: 4, ..., 6번 문제 정답 2)

3 3 3 3 4 4 # 문제별 배점 (1번 문제 점수: 3, 2번 문제 점수: 3, ..., 6번 문제 점수 4)

자료구조(수3, 금12)

1 4 3 4 1 2 # 제출한 답안지 (1번 문제: 1, 2번 문제: 4, ..., 6번 문제: 2)

출력 예

7 # 정답을 맞춘 문제: 2번, 6번

알고리즘 및 자료구조 설명

느낀 점

프로그램 코드

```
def solution():
    asw=input('문제별 정답을 입력하시오: ').split() #정답 입력
    score=input('문제별 배점을 입력하시오: ').split() #배점 입력
    submit=input('학생의 답을 입력하시오: ').split() #제출한 답안지 입력

    num=len(asw) #문제의 문항수
    result=[] #맞춘 문제의 수 기록
    for i in range(num):
        if asw[i]==submit[i]: #정답이 맞은 문제 문항을 기록
            result.append(i+1)
    answer=print(f'정답을 맞춘 문제: {result}번 입니다.')
    return answer

if __name__=="__main__":
```

```
sol=solution()
print(sol)
```

2-2번 문항 문제기술

2-2)

4지 선다형 문항들의 정답과 문항별 배점이 주어질 때, 제출한 n개의 답안지를 채점하여 최저 성적과 최고 성적을 출력하시오

입력 예

2 4 1 3 3 2 # 문항별 정답

3 3 3 3 4 4 # 문항별 배점

4 # 체출한 답안지 수 n

3 3 3 3 3 3 # 제출한 답안지1

2 2 2 2 2 2 # 제출한 답안지2

4 4 4 4 4 4 # 제출한 답안지3

2 4 1 3 3 2 # 제출한 답안지4

출력 예

3 20

알고리즘 및 자료구조 설명

느낀 점

프로그램 코드

```
def solution():
    asw=input('문제별 정답을 입력하시오: ').split() #정답 입력
    score=input('문제별 배점을 입력하시오: ').split() #배점 입력
    num_Paper=int(input("제출한 답안지의 갯수를 입력하시오: ")) #제출한 답안지 수
    submit=[None]*num_Paper#제출한 답안지의 빈 2 차 배열 list x 문제지 갯수

    for i in range(num_Paper):
        submit[i]=input('문제지 별 입력한 답을 기입하시오: ').split() #변수 i 로
답안을 입력받아 제출한 답안지 submit[]에 저장
        submit[i]=list(map(int,submit[i]))
    #list 들의 데이터 형태를 integer 로 변경
    asw=list(map(int,asw))
    score=list(map(int,score))

    result=[0]*num_Paper #맞춘 문제의 수 기록
    num_Test=len(asw) #문제의 문항수
    for i in range(num_Test):
        for k in range(num_Paper): #문제지 번호 측정용
            if submit[k][i] != asw[i]: #각 문제지 별로 답이 맞는 지 확인
                result[k]+=1*score[i] #맞은 문제별 배점을 고려한 점수 기록
    result=list(map(int,result))

    winner_Score=max(result)
    winner_Num=result.index(winner_Score)+1 #index 는 기본적으로 0 부터 n-
1 까지 이므로 제출한 시험지의 순서를 1~n 이라고 보았을 때, +1 처리

    return winner_Num , winner_Score

if __name__=="__main__":
    sol=solution()
    print(sol)
```

2-3번 문항 문제기술

2-3)

4지 선다형 문항들의 정답과 문항별 배점이 주어질 때, 제출한 n개의 답안지를 채점하여 (2) 정답률이 가장 낮은 문제의 번호를 출력하시오. 정답률이 가장 낮은 문제 여러 개일 경우 이 문제들의 번호를 오름차순으로 출력하시오.

입력 예

2 4 1 3 3 2 # 문항별 정답

3 3 3 3 4 4 # 문항별 배점

4 # 체출한 답안지 수 n

3 3 3 3 3 3 # 제출한 답안지1

2 2 2 2 2 2 # 제출한 답안지2

4 4 4 4 4 4 # 제출한 답안지3

2 4 1 3 3 2 # 제출한 답안지4

출력 예

1 2 4 5 6

알고리즘 및 자료구조 설명

느낀 점

프로그램 코드

3-1번 문항 문제기술

문제3)

3-1) 문자열이 회문이면 yes를 출력하고 그렇지 않으면 no를 출력하는 프로그램을 작성하시오. (대/소문자 구분하지 않음)

입력 예 1

absBa

출력 예 1

yes

입력 예 2

absaba

출력 예 2

no

알고리즘 및 자료구조 설명

느낀 점

프로그램 코드

```
def solution():
    n1=input("회문 측정할 문자열을 입력하시오: ")
    n1=n1.lower()#입력받은 문자열의 대/소문자를 '소문자' 형식으로 통일
    n2=list(n1) #문자열의 대/소문자 통일 후 list 형태로 n2 에 저장
    r_n=n2[::-1] #list 명[::-1] 명령어를 통해 손쉽게 list 내의 배열 변경가능

    check=0 #일치하는 문자열 확인
    answer="*"#결과
    for i in range(len(n2)):
        if n2[i]==r_n[i]: #회문 전 원소 vs 회문 후 원소 비교
            check+=1
    if check==len(n2): #모든 원소(회문 전, 후)가 check(일치)하는 지 확인
        answer="yes"
```

```
    else:
        answer="No"
    return answer

if __name__=="__main__":
    sol= solution()
    print(sol)
```

3-2번 문항 문제기술

3-2) 문자열 s의 서로 다른 모든 회문 부문자열(substring)을 사전식 순서대로 출력하는 프로그램을 작성하시오. 단, 대/소문자 구분하지 않으며, 출력하는 회문 부문자열의 각 문자는 소문자이다. sort 함수를 사용하여도 좋음)

입력 예 1

absAba

출력 예 2

a aba b

알고리즘 및 자료구조 설명

느낀 점

프로그램 코드