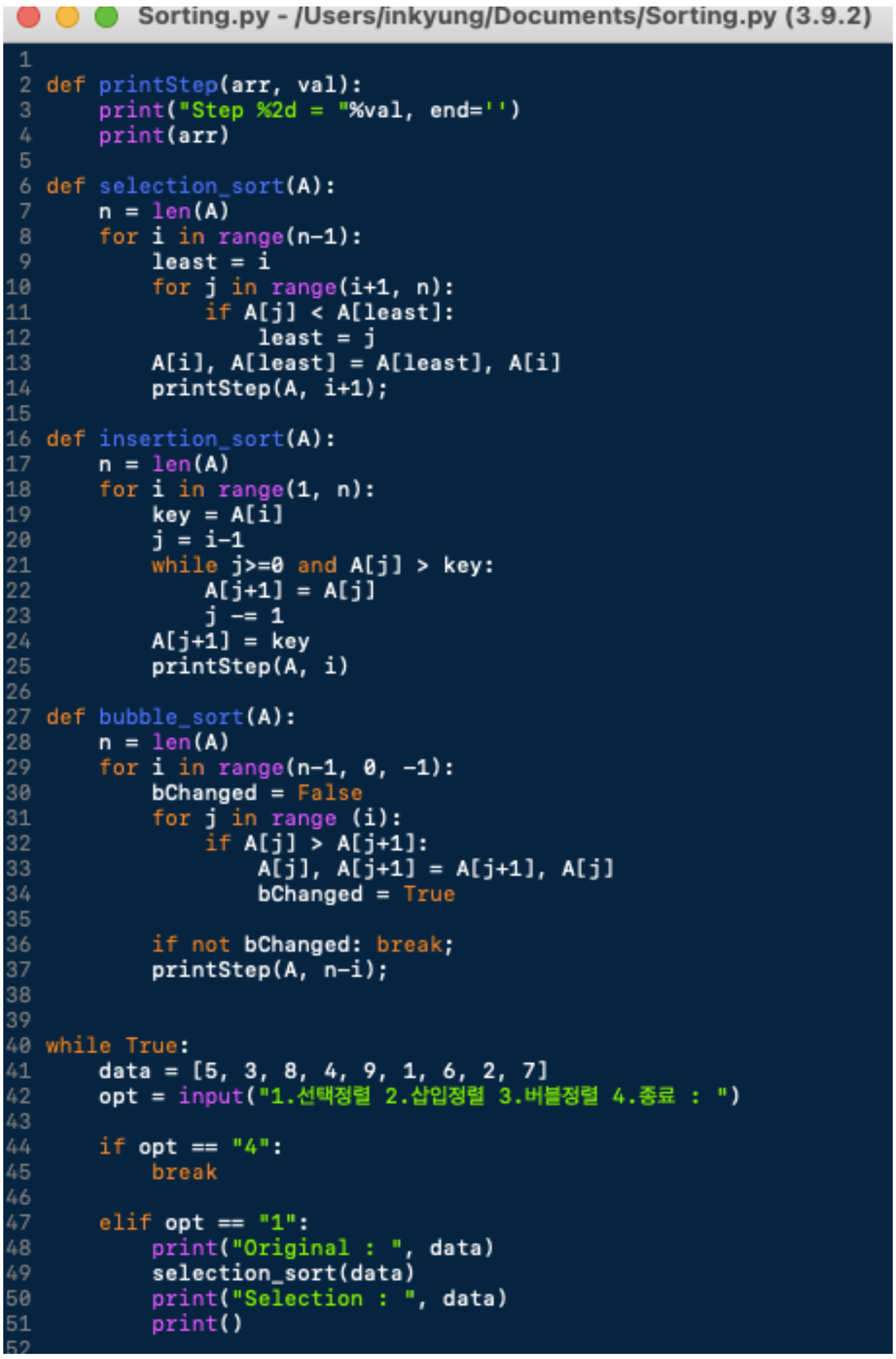
자료구조 7장 과제

Global Business & Technology
201904385 우인경

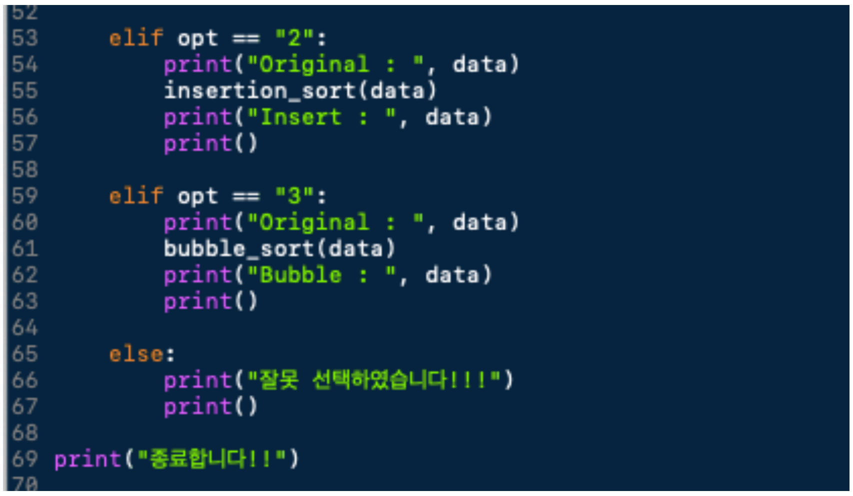
1. 선택정렬, 삽입정렬, 버블정렬

- 코드화면



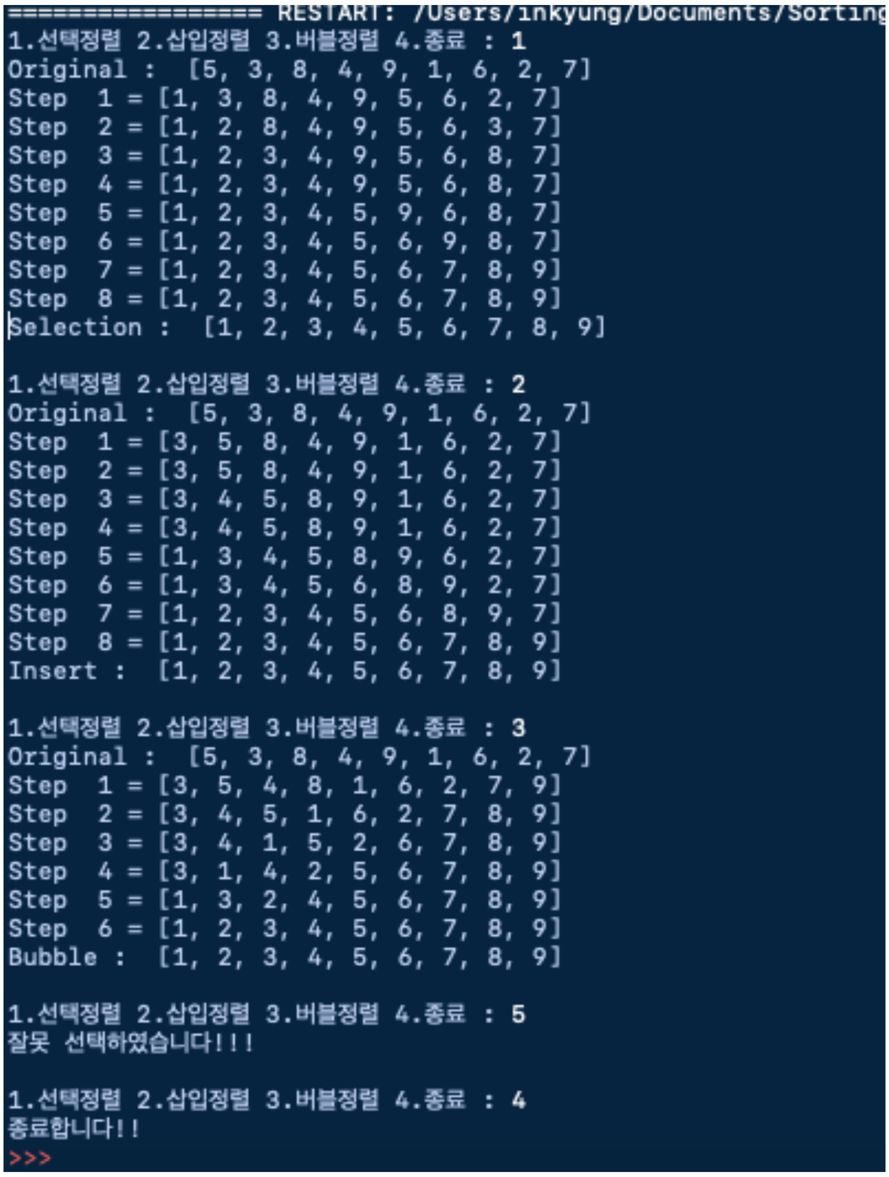
Sorting.py - /Users/inkyung/Documents/Sorting.py (3.9.2)

```python

def printStep(arr, val):
    print("Step %2d = "%val, end='')
    print(arr)

def selection_sort(A):
    n = len(A)
    for i in range(n-1):
        least = i
        for j in range(i+1, n):
            if A[j] < A[least]:
                least = j
        A[i], A[least] = A[least], A[i]
        printStep(A, i+1);

def insertion_sort(A):
    n = len(A)
    for i in range(1, n):
        key = A[i]
        j = i-1
        while j>=0 and A[j] > key:
            A[j+1] = A[j]
            j -= 1
        A[j+1] = key
        printStep(A, i)

def bubble_sort(A):
    n = len(A)
    for i in range(n-1, 0, -1):
        bChanged = False
        for j in range (i):
            if A[j] > A[j+1]:
                A[j], A[j+1] = A[j+1], A[j]
                bChanged = True

        if not bChanged: break;
        printStep(A, n-i);


while True:
    data = [5, 3, 8, 4, 9, 1, 6, 2, 7]
    opt = input("1.선택정렬 2.삽입정렬 3.버블정렬 4.종료 : ")

    if opt == "4":
        break

    elif opt == "1":
        print("Original : ", data)
        selection_sort(data)
        print("Selection : ", data)
        print()
```



```
    elif opt == "2":
        print("Original : ", data)
        insertion_sort(data)
        print("Insert : ", data)
        print()

    elif opt == "3":
        print("Original : ", data)
        bubble_sort(data)
        print("Bubble : ", data)
        print()

    else:
        print("잘못 선택하였습니다!!!")
        print()

print("종료합니다!!")
```

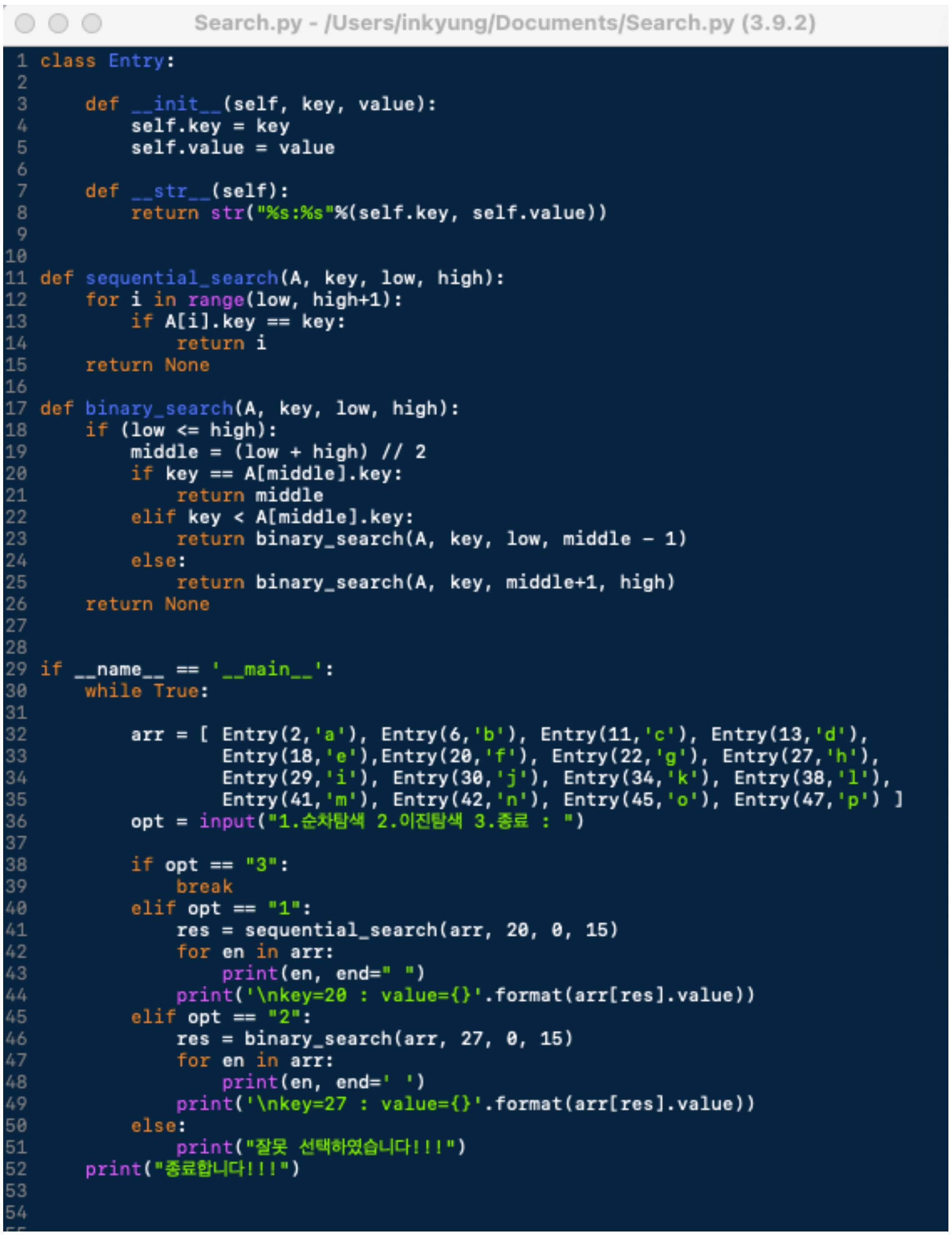
- 출력 화면



```
================= RESTART: /Users/inkyung/Documents/Sorting
1.선택정렬 2.삽입정렬 3.버블정렬 4.종료 : 1
Original :  [5, 3, 8, 4, 9, 1, 6, 2, 7]
Step  1 = [1, 3, 8, 4, 9, 5, 6, 2, 7]
Step  2 = [1, 2, 8, 4, 9, 5, 6, 3, 7]
Step  3 = [1, 2, 3, 4, 9, 5, 6, 8, 7]
Step  4 = [1, 2, 3, 4, 9, 5, 6, 8, 7]
Step  5 = [1, 2, 3, 4, 5, 9, 6, 8, 7]
Step  6 = [1, 2, 3, 4, 5, 6, 9, 8, 7]
Step  7 = [1, 2, 3, 4, 5, 6, 7, 8, 9]
Step  8 = [1, 2, 3, 4, 5, 6, 7, 8, 9]
Selection :  [1, 2, 3, 4, 5, 6, 7, 8, 9]

1.선택정렬 2.삽입정렬 3.버블정렬 4.종료 : 2
Original :  [5, 3, 8, 4, 9, 1, 6, 2, 7]
Step  1 = [3, 5, 8, 4, 9, 1, 6, 2, 7]
Step  2 = [3, 5, 8, 4, 9, 1, 6, 2, 7]
Step  3 = [3, 4, 5, 8, 9, 1, 6, 2, 7]
Step  4 = [3, 4, 5, 8, 9, 1, 6, 2, 7]
Step  5 = [1, 3, 4, 5, 8, 9, 6, 2, 7]
Step  6 = [1, 3, 4, 5, 6, 8, 9, 2, 7]
Step  7 = [1, 2, 3, 4, 5, 6, 8, 9, 7]
Step  8 = [1, 2, 3, 4, 5, 6, 7, 8, 9]
Insert :  [1, 2, 3, 4, 5, 6, 7, 8, 9]

1.선택정렬 2.삽입정렬 3.버블정렬 4.종료 : 3
Original :  [5, 3, 8, 4, 9, 1, 6, 2, 7]
Step  1 = [3, 5, 4, 8, 1, 6, 2, 7, 9]
Step  2 = [3, 4, 5, 1, 6, 2, 7, 8, 9]
Step  3 = [3, 4, 1, 5, 2, 6, 7, 8, 9]
Step  4 = [3, 1, 4, 2, 5, 6, 7, 8, 9]
Step  5 = [1, 3, 2, 4, 5, 6, 7, 8, 9]
Step  6 = [1, 2, 3, 4, 5, 6, 7, 8, 9]
Bubble :  [1, 2, 3, 4, 5, 6, 7, 8, 9]

1.선택정렬 2.삽입정렬 3.버블정렬 4.종료 : 5
잘못 선택하였습니다!!!

1.선택정렬 2.삽입정렬 3.버블정렬 4.종료 : 4
종료합니다!!
>>>
```

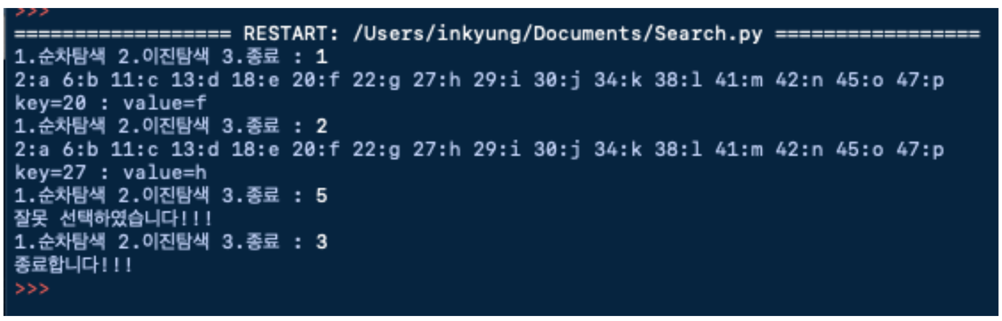
2. 순차탐색, 이진탐색

- 코드화면


Search.py - /Users/inkyung/Documents/Search.py (3.9.2)

```python
class Entry:

    def __init__(self, key, value):
        self.key = key
        self.value = value

    def __str__(self):
        return str("%s:%s"%(self.key, self.value))


def sequential_search(A, key, low, high):
    for i in range(low, high+1):
        if A[i].key == key:
            return i
    return None

def binary_search(A, key, low, high):
    if (low <= high):
        middle = (low + high) // 2
        if key == A[middle].key:
            return middle
        elif key < A[middle].key:
            return binary_search(A, key, low, middle - 1)
        else:
            return binary_search(A, key, middle+1, high)
    return None


if __name__ == '__main__':
    while True:

        arr = [ Entry(2,'a'), Entry(6,'b'), Entry(11,'c'), Entry(13,'d'),
                Entry(18,'e'),Entry(20,'f'), Entry(22,'g'), Entry(27,'h'),
                Entry(29,'i'), Entry(30,'j'), Entry(34,'k'), Entry(38,'l'),
                Entry(41,'m'), Entry(42,'n'), Entry(45,'o'), Entry(47,'p') ]
        opt = input("1.순차탐색 2.이진탐색 3.종료 : ")

        if opt == "3":
            break
        elif opt == "1":
            res = sequential_search(arr, 20, 0, 15)
            for en in arr:
                print(en, end=" ")
            print('\nkey=20 : value={}'.format(arr[res].value))
        elif opt == "2":
            res = binary_search(arr, 27, 0, 15)
            for en in arr:
                print(en, end=' ')
            print('\nkey=27 : value={}'.format(arr[res].value))
        else:
            print("잘못 선택하였습니다!!!")
    print("종료합니다!!!")
```

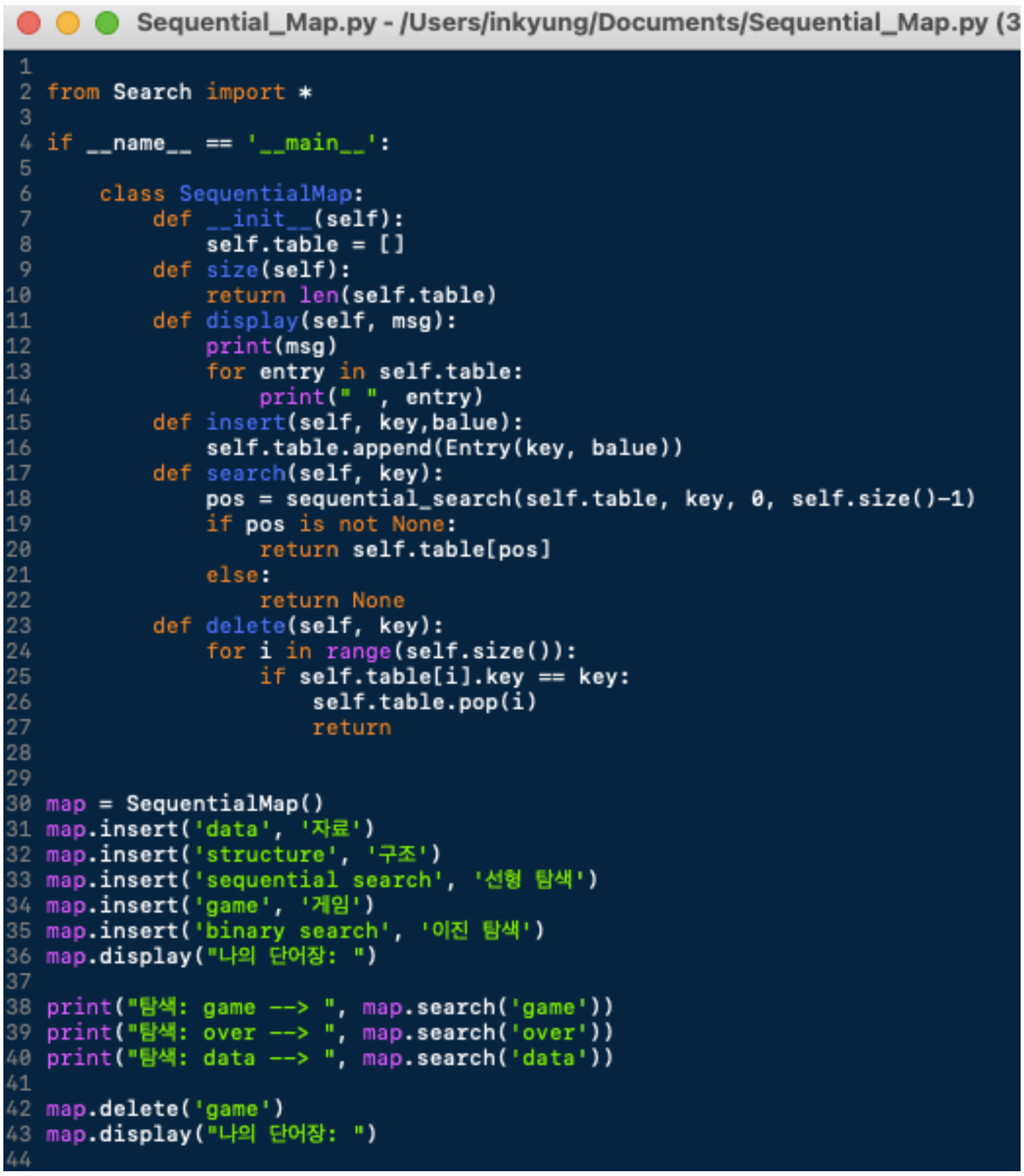
- 실행화면



```
>>>
================== RESTART: /Users/inkyung/Documents/Search.py ==================
1.순차탐색 2.이진탐색 3.종료 : 1
2:a 6:b 11:c 13:d 18:e 20:f 22:g 27:h 29:i 30:j 34:k 38:l 41:m 42:n 45:o 47:p
key=20 : value=f
1.순차탐색 2.이진탐색 3.종료 : 2
2:a 6:b 11:c 13:d 18:e 20:f 22:g 27:h 29:i 30:j 34:k 38:l 41:m 42:n 45:o 47:p
key=27 : value=h
1.순차탐색 2.이진탐색 3.종료 : 5
잘못 선택하였습니다!!!
1.순차탐색 2.이진탐색 3.종료 : 3
종료합니다!!!
>>>
```

3. 리스트를 이용한 순차탐색 맵 구현 및 실행

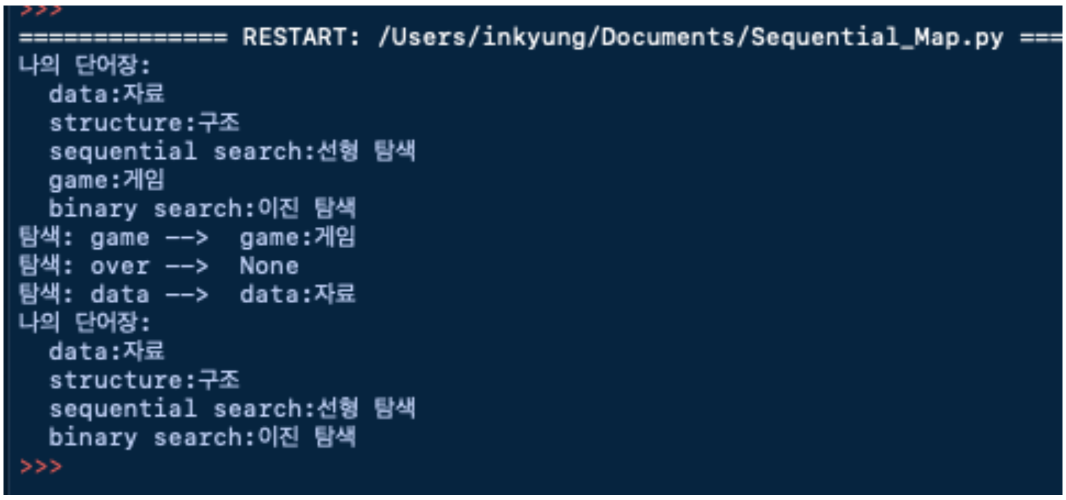
- 코드화면



Sequential_Map.py - /Users/inkyung/Documents/Sequential_Map.py (3

```python

from Search import *

if __name__ == '__main__':

    class SequentialMap:
        def __init__(self):
            self.table = []
        def size(self):
            return len(self.table)
        def display(self, msg):
            print(msg)
            for entry in self.table:
                print(" ", entry)
        def insert(self, key,balue):
            self.table.append(Entry(key, balue))
        def search(self, key):
            pos = sequential_search(self.table, key, 0, self.size()-1)
            if pos is not None:
                return self.table[pos]
            else:
                return None
        def delete(self, key):
            for i in range(self.size()):
                if self.table[i].key == key:
                    self.table.pop(i)
                    return


map = SequentialMap()
map.insert('data', '자료')
map.insert('structure', '구조')
map.insert('sequential search', '선형 탐색')
map.insert('game', '게임')
map.insert('binary search', '이진 탐색')
map.display("나의 단어장: ")

print("탐색: game --> ", map.search('game'))
print("탐색: over --> ", map.search('over'))
print("탐색: data --> ", map.search('data'))

map.delete('game')
map.display("나의 단어장: ")

```

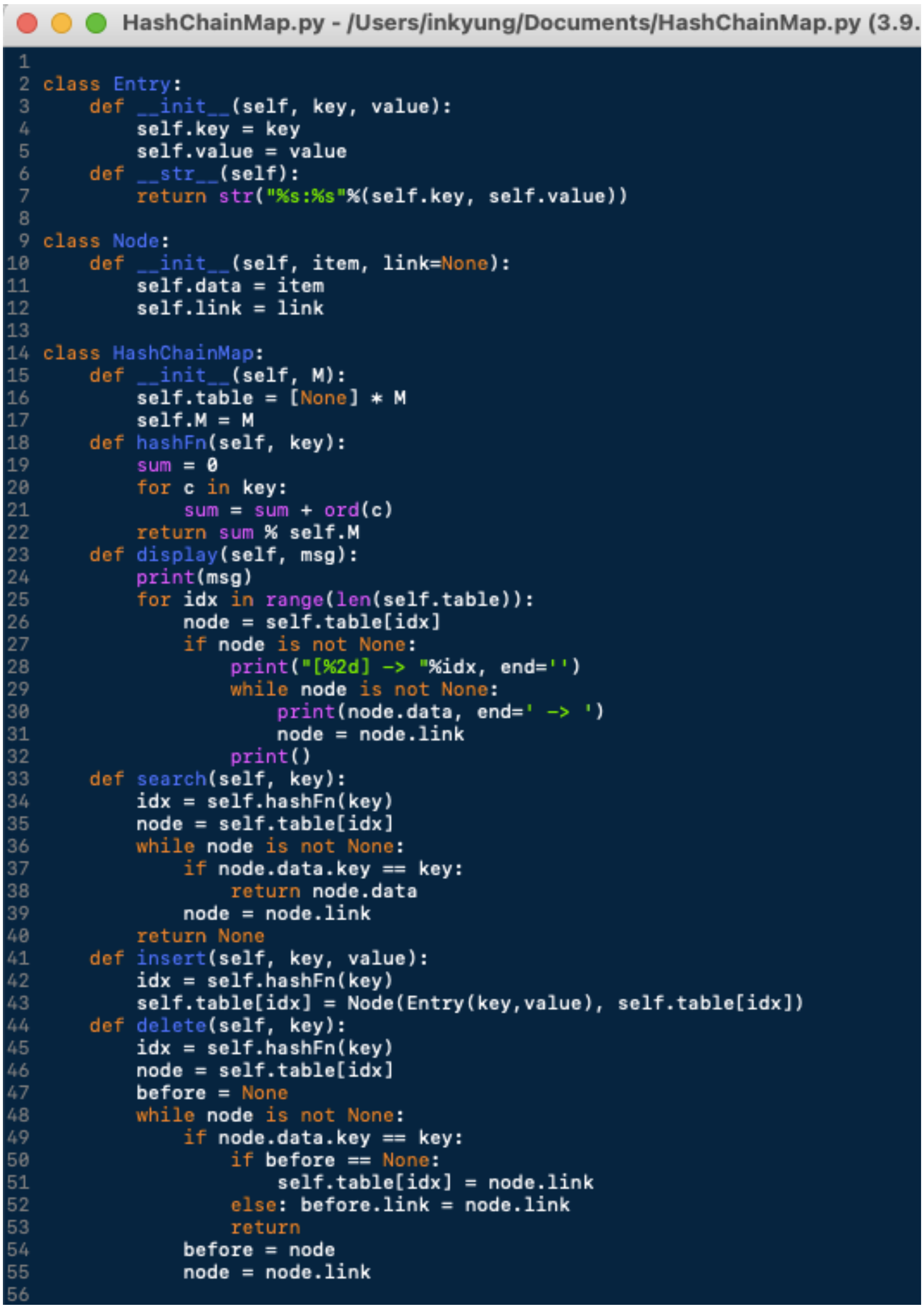
- 실행화면



```
>>>
============== RESTART: /Users/inkyung/Documents/Sequential_Map.py ===
나의 단어장:
  data:자료
  structure:구조
  sequential search:선형 탐색
  game:게임
  binary search:이진 탐색
탐색: game -->  game:게임
탐색: over -->  None
탐색: data -->  data:자료
나의 단어장:
  data:자료
  structure:구조
  sequential search:선형 탐색
  binary search:이진 탐색
>>>
```

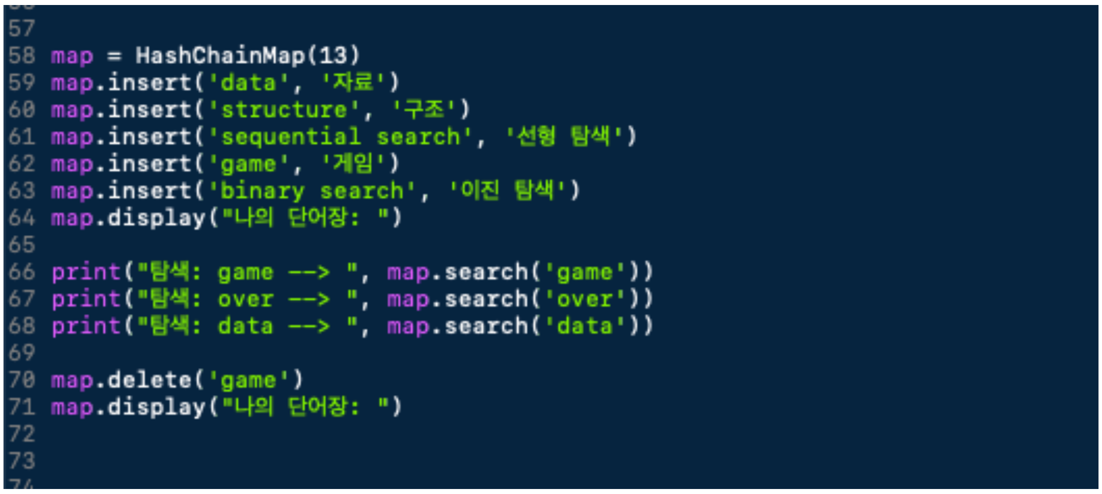
4. 체이닝을 이용한 해시 맵 구현 및 실행

- 코드화면



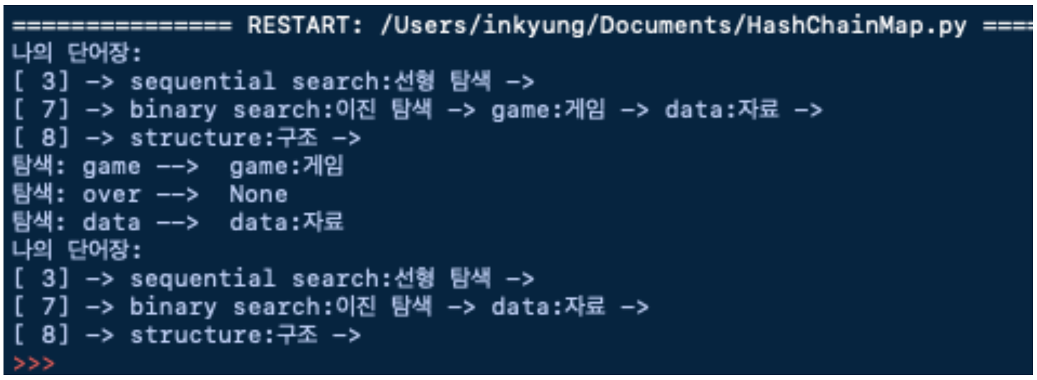
HashChainMap.py - /Users/inkyung/Documents/HashChainMap.py (3.9.

```python

class Entry:
    def __init__(self, key, value):
        self.key = key
        self.value = value
    def __str__(self):
        return str("%s:%s"%(self.key, self.value))

class Node:
    def __init__(self, item, link=None):
        self.data = item
        self.link = link

class HashChainMap:
    def __init__(self, M):
        self.table = [None] * M
        self.M = M
    def hashFn(self, key):
        sum = 0
        for c in key:
            sum = sum + ord(c)
        return sum % self.M
    def display(self, msg):
        print(msg)
        for idx in range(len(self.table)):
            node = self.table[idx]
            if node is not None:
                print("[%2d] -> "%idx, end='')
                while node is not None:
                    print(node.data, end=' -> ')
                    node = node.link
                print()
    def search(self, key):
        idx = self.hashFn(key)
        node = self.table[idx]
        while node is not None:
            if node.data.key == key:
                return node.data
            node = node.link
        return None
    def insert(self, key, value):
        idx = self.hashFn(key)
        self.table[idx] = Node(Entry(key,value), self.table[idx])
    def delete(self, key):
        idx = self.hashFn(key)
        node = self.table[idx]
        before = None
        while node is not None:
            if node.data.key == key:
                if before == None:
                    self.table[idx] = node.link
                else: before.link = node.link
                return
            before = node
            node = node.link

```



```
map = HashChainMap(13)
map.insert('data', '자료')
map.insert('structure', '구조')
map.insert('sequential search', '선형 탐색')
map.insert('game', '게임')
map.insert('binary search', '이진 탐색')
map.display("나의 단어장: ")

print("탐색: game --> ", map.search('game'))
print("탐색: over --> ", map.search('over'))
print("탐색: data --> ", map.search('data'))

map.delete('game')
map.display("나의 단어장: ")
```

- 출력 화면



```
=============== RESTART: /Users/inkyung/Documents/HashChainMap.py ====
나의 단어장:
[ 3] -> sequential search:선형 탐색 ->
[ 7] -> binary search:이진 탐색 -> game:게임 -> data:자료 ->
[ 8] -> structure:구조 ->
탐색: game -->  game:게임
탐색: over -->  None
탐색: data -->  data:자료
나의 단어장:
[ 3] -> sequential search:선형 탐색 ->
[ 7] -> binary search:이진 탐색 -> data:자료 ->
[ 8] -> structure:구조 ->
>>>
```