ชื่อ – นามสกุล **วัฒชัย เตชะลือ** รหัสนักศึกษา **65015137**

Data Structures and Algorithm

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

การทดลองที่ 8 : เปรียบเทียบการทำงานของ sequential search และ binary search

จุดประสงค์

1. นักศึกษาเข้าใจการทำงานและผลลัพธ์ของ sequential search และ binary search พร้อมทั้งสามารถเปรียบเทียบได้
2. นักศึกษาเข้าใจข้อจำกัดของ sequential search และ binary search ที่นำไปใช้ในงานต่างๆ

โปรแกรมตัวอย่าง

```
import random
comparecount = 0
def binary_search(arr, low, high, x):
 global comparecount
 if high >= low:
  mid = (high + low) // 2
  if arr[mid] == x:
   return mid
  elif arr[mid] > x:
   comparecount +=1
   print("comparecount = ",comparecount, "low = ",low, "high =",mid-1)
   return binary_search(arr, low, mid - 1, x)
  else:
   comparecount +=1
   print("comparecount = ",comparecount, "low = ",mid+1, "high =",high)
   return binary_search(arr, mid + 1 , high, x)
 else:
  return -1

def sequential_search(arr,x):
 global comparecount
 for i in arr:
  comparecount +=1
  if i==x:
   return comparecount
 return -1

datcount = 100000
## incase of sequential search
#arr = [random.randint(1,10000000) for i in range(datcount)]

## in case of binary search
#arr = sorted([random.randint(1,10000000) for i in range(datcount)])

## in case of succesfully search
#x = arr[random.randint(1,datcount)]
```

ชื่อ – นามสกุล **วัฒชัย เตชะลือ** รหัสนักศึกษา **65015137**

```
## in case of unsuccessfully search
#x = -1

## worstcase successfully search
#x = arr[-1]

print("key =",x)
print("data len = ",len(arr))

## in case of binary search
#result = binary_search(arr, 0, len(arr)-1, x)
## incase of sequential search
#result = sequential_search(arr,x)

print("compare count = " ,comparecount)
if result != -1:
        print("Element is present at index", str(result))
else:
        print("Element is not present in array")
```

จากโปรแกรมตัวอย่าง เป็นโปรแกรมสำหรับการทดสอบ sequential search และ binary search ใน 3 กรณี คือกรณีที่ค้นหาเจอแบบทั่วไป กรณีที่ค้นหาเจอแบบแย่ที่สุด และกรณีที่ค้นหาไม่เจอข้อมูลที่ต้องการ โดยจะต้องมีการปรับแต่งโปรแกรมโดย uncomment บรรทัดที่ต้องใช้งานให้ถูกต้อง โดยจุดสำคัญคือข้อมูลที่ใช้สำหรับ binary search ต้องเป็นข้อมูลที่ทำการเรียงลำดับแล้ว และข้อมูลที่ใช้สำหรับ sequential search ไม่จำเป็นต้องเรียงลำดับก็ได้

สำหรับการทดลองนี้จะใช้ขนาดข้อมูลคงที่คือ 100,000 ชุดข้อมูล และใช้การทดลองซ้ำๆ เพื่อหาค่าเฉลี่ยของการทำงานพื้นฐาน โดยนับที่จำนวนของการเปรียบเทียบข้อมูลอ้างอิง กับข้อมูลที่ต้องการค้นหา หากมีจำนวนการเปรียบเทียบน้อยกว่าจะถือว่ามีการทำงานที่รวดเร็วกว่า

ตอนที่ 1 : การทำงานของ Sequential Search

Successfully Search , Average Case :

1. ให้นักศึกษาทดลองโปรแกรมที่กำหนดให้ โดยกำหนดให้การทำงานเป็น sequential search ที่ข้อมูล 100,000 ชุดข้อมูล โดยกำหนด key ที่จะค้นหาเป็นแบบสุ่มตำแหน่งให้สามารถค้นหาเจอ
2. ทำการทดลองรันโปรแกรมทั้งหมด 10 ครั้งแล้วหาค่าเฉลี่ยของจำนวนการค้นหาทั้งหมด

ครั้งที่	จำนวนครั้งที่เทียบข้อมูล
1	89310
2	61036
3	96262
4	77898
5	99287
6	92246
7	42815
8	5016
9	66476

10	14782
ค่าเฉลี่ย	64512.8

ค่าเฉลี่ย / จำนวนชุดข้อมูลทั้งหมด = 10

Successfully Search , Worst Case :

1. ให้นักศึกษาทดลองโปรแกรมที่กำหนดให้ โดยกำหนดให้การทำงานเป็น sequential search ที่ข้อมูล 100,000 ชุดข้อมูล โดยกำหนด key ที่จะค้นหาเป็นข้อมูลตัวสุดท้ายของรายการ
2. ทำการทดลองรันโปรแกรมทั้งหมด 10 ครั้งแล้วหาค่าเฉลี่ยของจำนวนการค้นหาทั้งหมด

ครั้งที่	จำนวนครั้งที่เทียบข้อมูล
1	100000
2	100000
3	100000
4	100000
5	100000
6	100000
7	100000
8	100000
9	100000
10	100000
ค่าเฉลี่ย	100000

ค่าเฉลี่ย / จำนวนชุดข้อมูลทั้งหมด = **100000**

Unsuccessfully Search :

1. ให้นักศึกษาทดลองโปรแกรมที่กำหนดให้ โดยกำหนดให้การทำงานเป็น sequential search ที่ข้อมูล 100,000 ชุดข้อมูล โดยกำหนด key ที่จะค้นหาเป็นข้อมูลที่ไม่อยู่ในรายการ
2. ทำการทดลองรันโปรแกรมทั้งหมด 10 ครั้งแล้วหาค่าเฉลี่ยของจำนวนการค้นหาทั้งหมด

ครั้งที่	จำนวนครั้งที่เทียบข้อมูล
1	100000
2	100000
3	100000
4	100000
5	100000
6	100000
7	100000
8	100000
9	100000
10	100000
ค่าเฉลี่ย	100000

ค่าเฉลี่ย / จำนวนชุดข้อมูลทั้งหมด = **100000**

ชื่อ – นามสกุล **วัฒชัย เตชะลือ** รหัสนักศึกษา **65015137**

ตอนที่ 2 : การทำงานของ Binary Search

Successfully Search , Average Case :

1. ให้นักศึกษาทดลองโปรแกรมที่กำหนดให้ โดยกำหนดให้การทำงานเป็น binary search ที่ข้อมูล 100,000 ชุดข้อมูล โดยกำหนด key ที่จะค้นหาเป็นแบบสุ่มตำแหน่งให้สามารถค้นหาเจอ
2. ทำการทดลองรันโปรแกรมทั้งหมด 10 ครั้งแล้วหาค่าเฉลี่ยของจำนวนการค้นหาทั้งหมด

ครั้งที่	จำนวนครั้งที่เทียบข้อมูล
1	15
2	15
3	16
4	15
5	15
6	15
7	15
8	16
9	12
10	15
ค่าเฉลี่ย	14.9

ค่าเฉลี่ย / จำนวนชุดข้อมูลทั้งหมด = **14.9**

Successfully Search , Worst Case :

1. ให้นักศึกษาทดลองโปรแกรมที่กำหนดให้ โดยกำหนดให้การทำงานเป็น binary search ที่ข้อมูล 100,000 ชุดข้อมูล โดยกำหนด key ที่จะค้นหาเป็นข้อมูลตัวสุดท้ายของรายการ
2. ทำการทดลองรันโปรแกรมทั้งหมด 10 ครั้งแล้วหาค่าเฉลี่ยของจำนวนการค้นหาทั้งหมด

ครั้งที่	จำนวนครั้งที่เทียบข้อมูล
1	16
2	16
3	16
4	16
5	16
6	16
7	16
8	16
9	16
10	16
ค่าเฉลี่ย	16

ค่าเฉลี่ย / จำนวนชุดข้อมูลทั้งหมด = **16**

Unsuccessfully Search :

1. ให้นักศึกษาทดลองโปรแกรมที่กำหนดให้ โดยกำหนดให้การทำงานเป็น binary search ที่ข้อมูล 100,000 ชุดข้อมูล โดยกำหนด key ที่จะค้นหาเป็นข้อมูลที่ไม่อยู่ในรายการ
2. ทำการทดลองรันโปรแกรมทั้งหมด 10 ครั้งแล้วหาค่าเฉลี่ยของจำนวนการค้นหาทั้งหมด

ครั้งที่	จำนวนครั้งที่เทียบข้อมูล
1	16

2	16
3	16
4	16
5	16
6	16
7	16
8	16
9	16
10	16
ค่าเฉลี่ย	16

ค่าเฉลี่ย / จำนวนชุดข้อมูลทั้งหมด = **16**

ตอนที่ 3 : ตอบคำถามและวิเคราะห์การทำงาน

1. Binary search มีข้อจำกัดอย่างไรบ้าง

ตอบ ข้อจำกัดของ binary search คือ ข้อมูลต้องเรียงลำดับก่อน ไม่งั้นจะหาไม่เจอ **binary search** ทำงานโดยการแบ่งข้อมูลออกเป็นครึ่งๆ แล้วเปรียบเทียบค่าเป้าหมายกับค่ากลางของข้อมูล ถ้าค่าเป้าหมายอยู่ตรงกลางก็เจอเลย แต่ถ้าไม่อยู่ตรงกลางก็ต้องแบ่งข้อมูลออกเป็นครึ่งๆ อีกครั้ง แล้วเปรียบเทียบค่าเป้าหมายกับค่ากลางของข้อมูลในครึ่งนั้นๆ ทำแบบนี้ไปเรื่อยๆ จนกว่าค่าเป้าหมายจะอยู่ตรงกลาง หรือจนกว่าข้อมูลจะเหลือเพียงตัวเดียว แล้วถ้าข้อมูลไม่เรียงลำดับ ขั้นตอนนี้จะไม่สามารถทำงานได้ เพราะไม่รู้ว่าค่าเป้าหมายอยู่ตรงไหน ก็เลยไม่สามารถแบ่งข้อมูลออกเป็นครึ่งๆ ได้

สรุปง่ายๆ คือ binary search ต้องจัดเรียงข้อมูลก่อน ไม่งั้นจะหาไม่เจอครับ

2. ถ้าเป็นข้อมูลที่ไม่มีการเรียงลำดับ ใน binary search นักศึกษาคิดว่าจะเกิดผลการทำงานเป็นอย่างไร

ตอบ ถ้าข้อมูลมีน้อยๆ ก็ไม่มีปัญหาอะไร แต่ถ้าข้อมูลมีเยอะๆ การทำงานแบบ linear search จะใช้เวลาในการค้นหาเพิ่มขึ้นตามจำนวนข้อมูล กรณีที่ข้อมูลมีขนาดใหญ่มาก อาจใช้เวลานานมากในการค้นหาจนอาจทำให้ระบบทำงานช้าลงได้ครับ หรือ **binary search** กับข้อมูลที่ไม่เรียงลำดับ จะทำงานได้ช้ามาก ถ้าข้อมูลมีเยอะๆ ควรจัดเรียงข้อมูลก่อน ไม่งั้นจะหาไม่เจอเลย

3. จากจำนวนข้อมูลที่เท่ากัน ใน worst case และกรณีที่ค้นหาไม่เจอ การทำงานของ search แบบไหนไวกว่ากัน

ตอบ Binary search

4. จากจำนวนข้อมูลที่เท่ากัน ในกรณีทั่วไป การทำงานของ search แบบไหนไวกว่ากัน

ตอบ Binary search

5. ในกรณีที่รายการข้อมูลมีการ update บ่อยๆ นักศึกษาคิดว่าการ search แบบใดทำงานได้ไวกว่ากัน ให้นักศึกษาลองให้เหตุผลสนับสนุนคำตอบของตนเอง และกำหนดวิธีการทดสอบ

ตอบ หลังจากที่ตอบข้อที่ 1,2,3,4 มาผมคิดว่ายังไงก็เป็น Binary search แน่นอนครับ เพราะว่า Big O ของ **Binary search** คือ Worst case: O(log n) และ กรณีที่ค้นหาไม่เจอ: O(log n) อีกอันคือ **Linear search** และ Worst case: O(n) และ กรณีที่ค้นหาไม่เจอ: O(n) ดังนั้น ถ้าข้อมูลมีการ update บ่อยๆ การ search แบบ binary search จะทำงานได้ไวกว่าการ search แบบ linear search เสมอ เพราะ binary search จะสามารถจำกัดขอบเขตของข้อมูลที่ต้องการค้นหาได้อย่างรวดเร็ว ในขณะที่ linear search จะต้องเปรียบเทียบค่าเป้าหมายกับข้อมูลทีละตัวจนกว่าจะเจอค่าเป้าหมายหรือหมดข้อมูลครับ