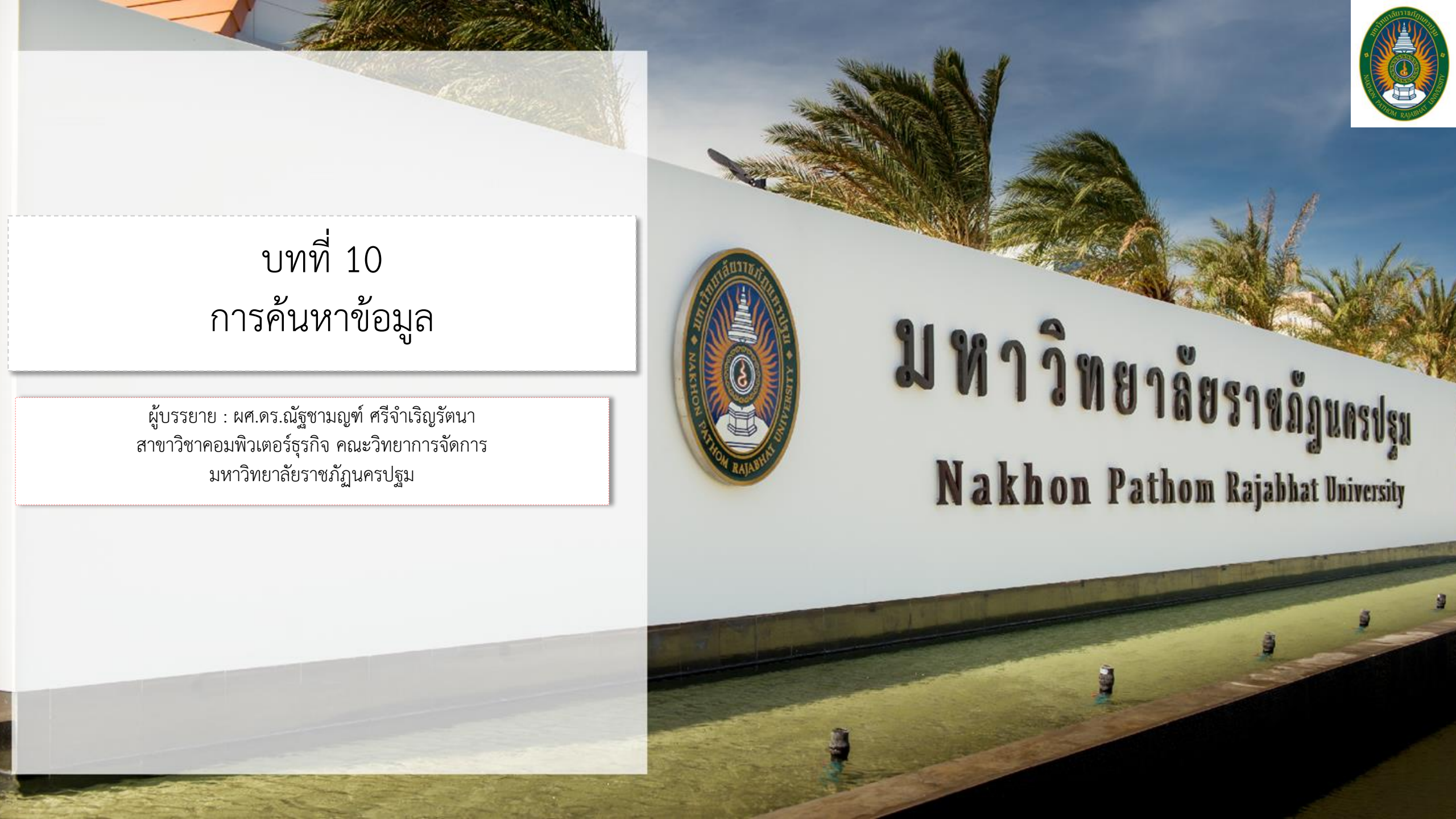
มหาวิทยาลัยราชภัฏนครปฐม



บทที่ 10
การค้นหาข้อมูล

ผู้บรรยาย : ผศ.ดร.ณัฐชามญฑ์ ศรีจำเริญรัตนา
สาขาวิชาคอมพิวเตอร์ธุรกิจ คณะวิทยาการจัดการ
มหาวิทยาลัยราชภัฏนครปฐม

การค้นหาข้อมูล

มีประโยชน์ของการค้นหาข้อมูลที่เก็บในระบบคอมพิวเตอร์

- การนำข้อมูลเพื่อแสดงผลหรือตรวจสอบรายละเอียดของข้อมูลก่อนนำมาใช้ประกอบการทำงาน
- การค้นหาข้อมูลเพื่อเปลี่ยนแปลงแก้ไขข้อมูลบางอย่างในข้อมูลที่ค้นหา
- อื่นๆ

การค้นหาข้อมูลสามารถตามลักษณะการค้นหาแบ่งเป็น 2 ประเภท

- **การค้นหาแบบลำดับ** หรือเรียกว่าการค้นหาข้อมูลแบบเชิงเส้น เป็นการค้นหาข้อมูลที่ค้นหาแบบต่อเนื่องไม่ตามลำดับ การจัดเก็บข้อมูลเพื่อนำมาเข้ากระบวนการค้นหาเป็นได้ทั้งข้อมูลที่เรียงลำดับอยู่แล้วหรืออาจไม่เรียงอยู่ก็เป็นไปได้
- **การค้นหาแบบไบนารี** เป็นโครงสร้างการทำงานกับข้อมูลที่เรียงลำดับอยู่แล้ว และต้องรู้จำนวนข้อมูลทั้งหมด และเหมาะกับการทำงานกับข้อมูลที่มีจำนวนมาก หลักการทำงานของไบนารีเป็นการค้นหาข้อมูลแบบแบ่งข้อมูลเพื่อการทำงานทีละ 2 ส่วนย่อย เป็นส่วนที่คาดว่าจะพบและส่วนที่นอกเขตข้อมูล และทุก ๆ การทำงานจะทำงานในส่วนที่คิดว่าจะพบข้อมูลในทุก ๆ การแบ่งตลอดเวลา

10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

- การค้นหาข้อมูลแบบเรียงลำดับ เรียกอีกอย่างว่า linear search หรือ sequential search
- ใช้สำหรับค้นหาข้อมูลตามลำดับทุก ๆ เรคคอร์ดหรือทุกโหนดข้อมูล
  เหมาะสำหรับโครงสร้างข้อมูลที่ไม่เรียงลำดับ
- แต่ไม่นิยมใช้กับข้อมูลที่มีปริมาณมาก ๆ หรือข้อมูลที่ต้องใช้งานบ่อย ๆ เนื่องจากเทคนิคการค้นหาข้อมูลลักษณะนี้จะไม่มีประสิทธิภาพนั่นเอง

10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

อัลกอริทึมการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ สามารถอธิบายได้ดังนี้

1) กำหนดข้อมูล (search) ที่ต้องการค้นหา

2) เปรียบเทียบข้อมูลในข้อมูลลำดับทีละลำดับกับข้อมูลที่ต้องการค้นหา (search) ในข้อ 1

2.1) ถ้าพบข้อมูล แสดงข้อความ “SUCCESSFUL SEARCH! found at location ตำแหน่งที่พบ”

2.2) ถ้าไม่พบข้อมูลที่ต้องการค้นหา แสดงข้อความ “Search is FAILED ข้อมูลที่ค้นหา is not present in the list”

10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.1 โปรแกรมการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

```
1  #include <iostream>
2  using namespace std;
3
4  void display (int [], int);
5  void sequential_search (int[], int, int );
6  int size=10;
7  int found=0;
```

```
8  int main()
9  {
10    int   search ;
11    int data[] = {7, 12, 41, 48, 59, 61, 63,
      65, 74, 93};
12    cout << "Data for Search: ";
13    display (data, size);
14    cout <<"Enter the element
                to be searched :";
15    cin>>search;
16    sequential_search  (data, size, search);
17 }
18
```

10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.1 โปรแกรมการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

```
19  void sequential_search (int data[],
    int size, int search)
20  {
21      for (int i=0; i < size ; i++)
22      {
23          if(search == data[i] )
24          {
25              found = 1;
26              out << search <<"SUCCESSFUL
                SEARCH! found at location "<<i+1;
27              break;
28          }
29      }
```

```
30  if ( found != 1)
31  {
32    cout <<"Search is FAILED "<< search
        <<" is not present in the list.\n";
33    }
34  }
35
36  void display (int data[], int size)
37  {
38    for (int i = 0; i < size; ++i)
39    {
40      cout << data[i] << " ";
41    }
42    cout << endl;
43  }
```

10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ผลการทำงานของโปรแกรม

กำหนดให้ค้นหาข้อมูล 48 ในชุดข้อมูลที่กำหนด

กรณีค้นพบข้อมูล

```
Data for Search: 7 12 41 48 59 61 63 65 74 93
Enter the element to be searched :48
48 SUCCESSFUL SEARCH! found at location 4
```

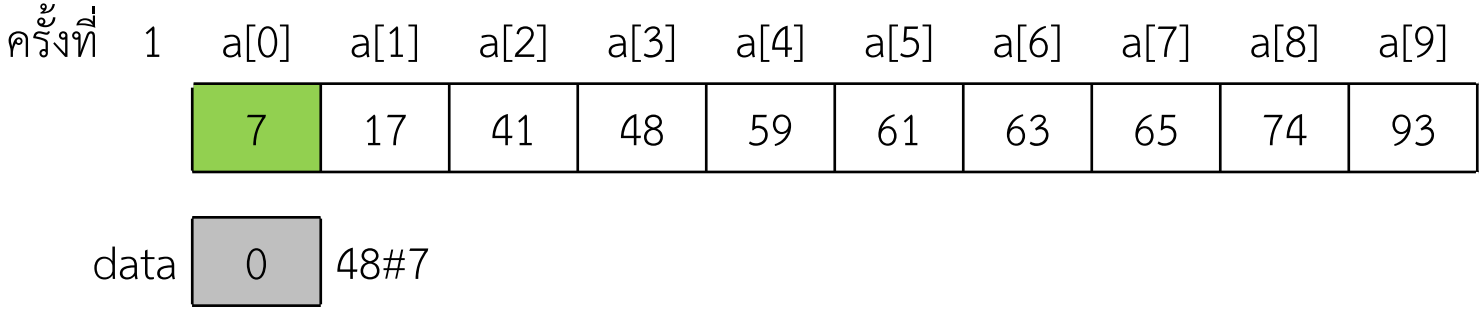
กรณีค้นไม่พบข้อมูล

```
Data for Search: 7 12 41 48 59 61 63 65 74 93
Enter the element to be searched :46
Search is FAILED 46 is not present in the list.
```

10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.2 จงแสดงขั้นตอนการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

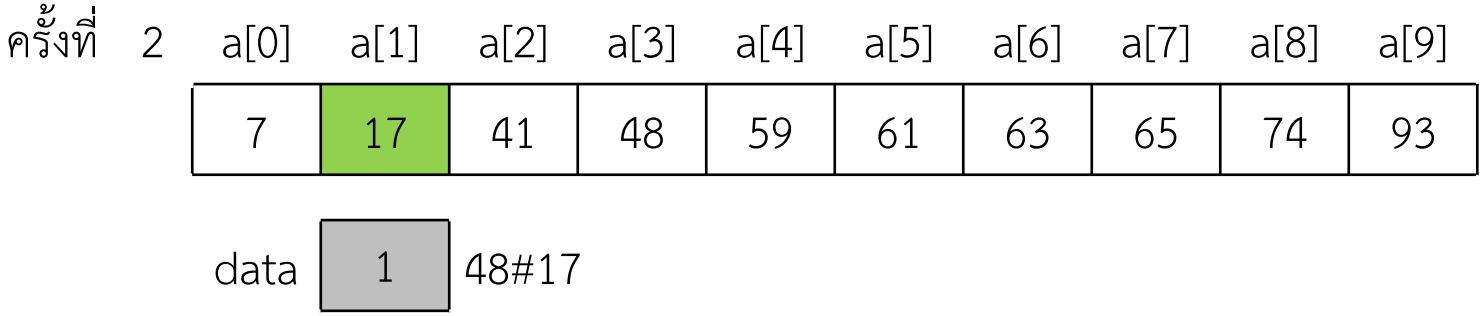
- เพื่อค้นหาค่า 48 กรณีค้นพบข้อมูล



10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.2 จงแสดงขั้นตอนการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

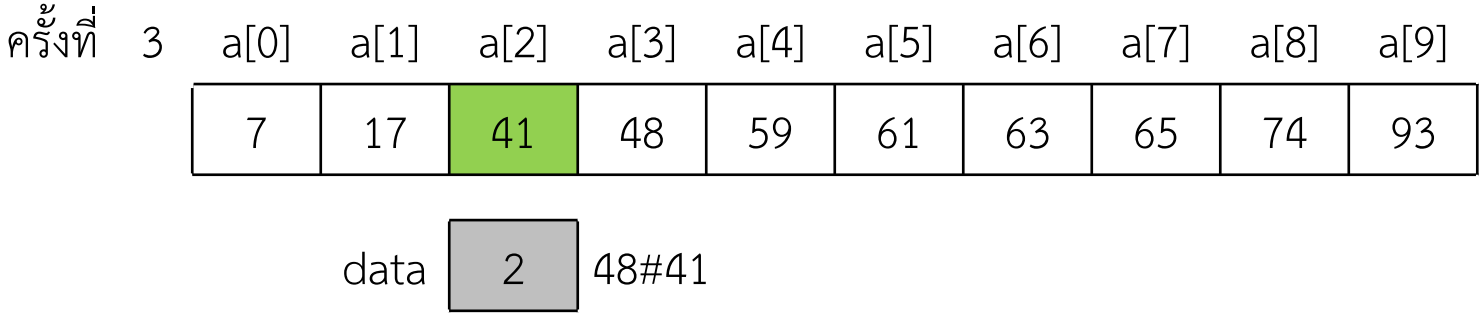
- เพื่อค้นหาค่า 48 กรณีค้นพบข้อมูล



10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.2 จงแสดงขั้นตอนการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ(ต่อ)

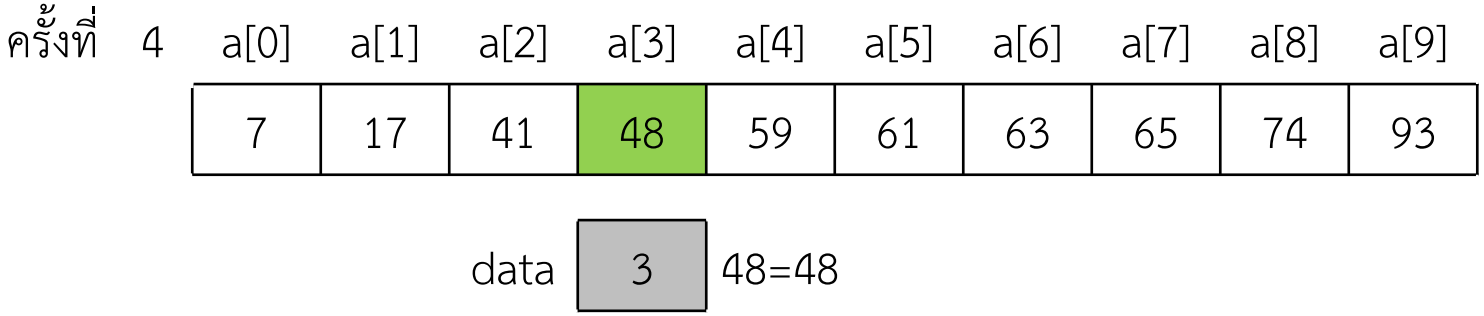
- เพื่อค้นหาค่า 48 กรณีค้นพบข้อมูล



10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.2 จงแสดงขั้นตอนการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ(ต่อ)

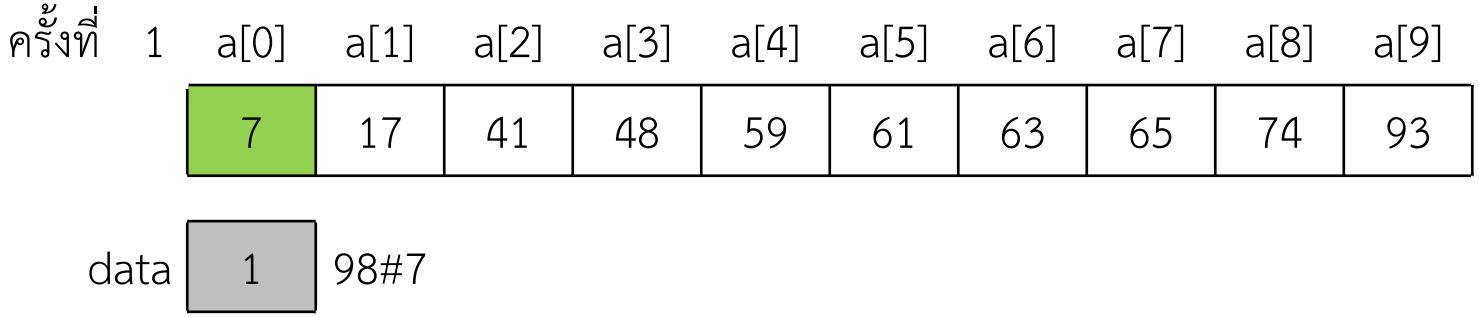
- เพื่อค้นหาค่า 48 กรณีค้นพบข้อมูล



10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.3 แสดงขั้นตอนการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

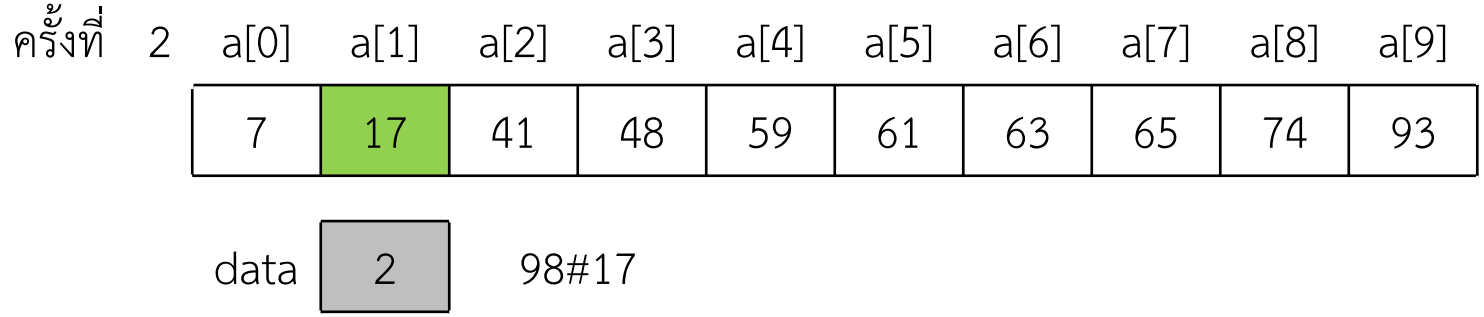
- เพื่อค้นหาค่า 98 กรณีไม่พบข้อมูลที่ต้องการ



10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.3 แสดงขั้นตอนการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

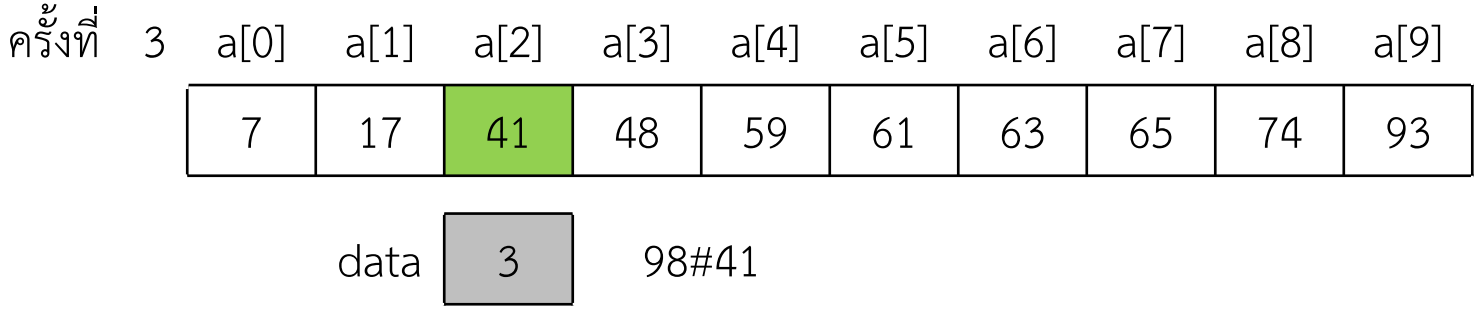
- เพื่อค้นหาค่า 98 กรณีไม่พบข้อมูลที่ต้องการ



10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.3 แสดงขั้นตอนการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ(ต่อ)

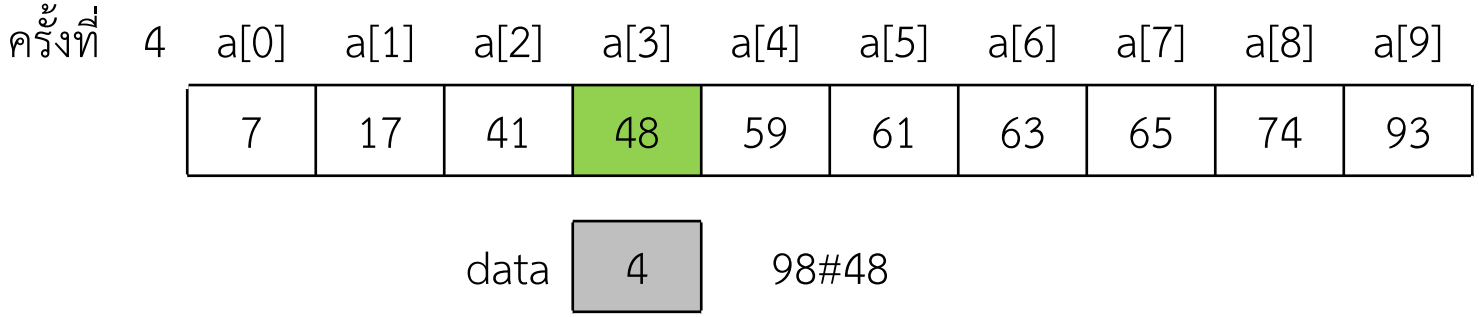
- เพื่อค้นหาค่า 98 กรณีไม่พบข้อมูลที่ต้องการ



10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.3 แสดงขั้นตอนการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ(ต่อ)

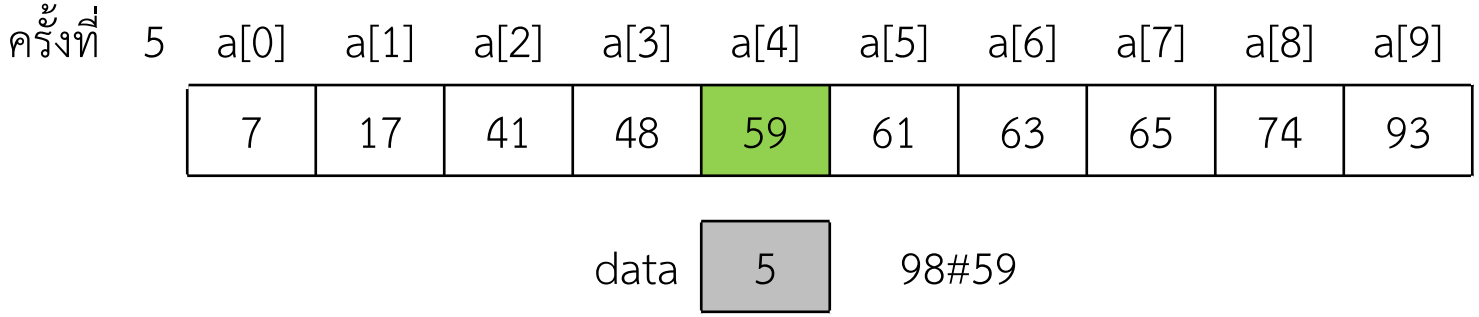
- เพื่อค้นหาค่า 98 กรณีไม่พบข้อมูลที่ต้องการ



10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.3 แสดงขั้นตอนการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ(ต่อ)

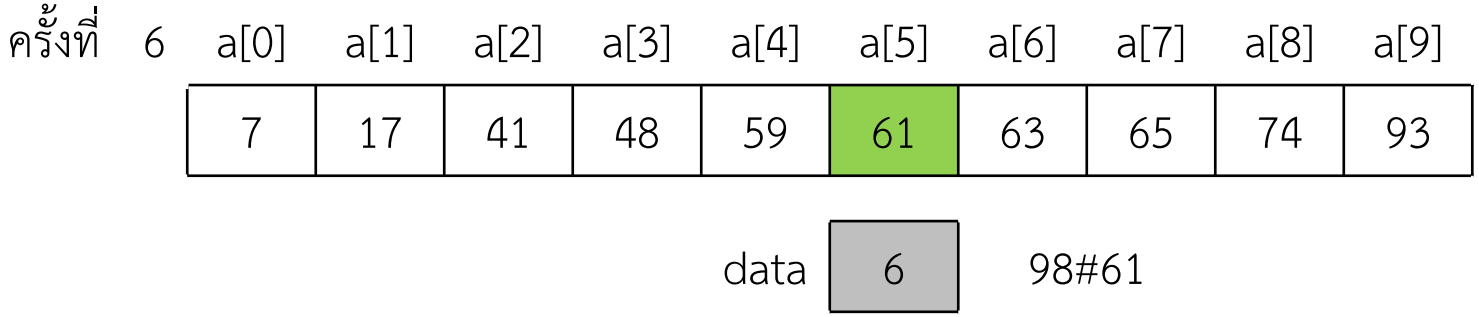
- เพื่อค้นหาค่า 98 กรณีไม่พบข้อมูลที่ต้องการ



10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.3 แสดงขั้นตอนการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ(ต่อ)

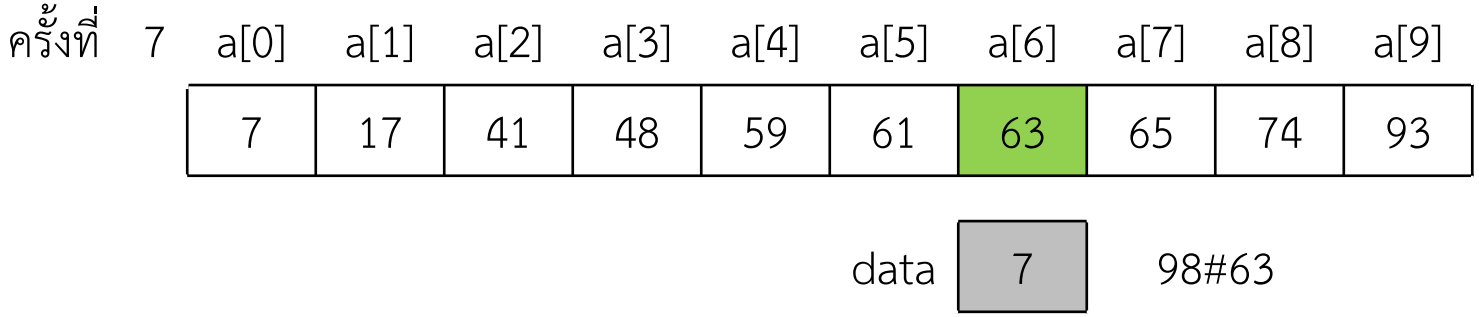
- เพื่อค้นหาค่า 98 กรณีไม่พบข้อมูลที่ต้องการ



10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.3 แสดงขั้นตอนการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ(ต่อ)

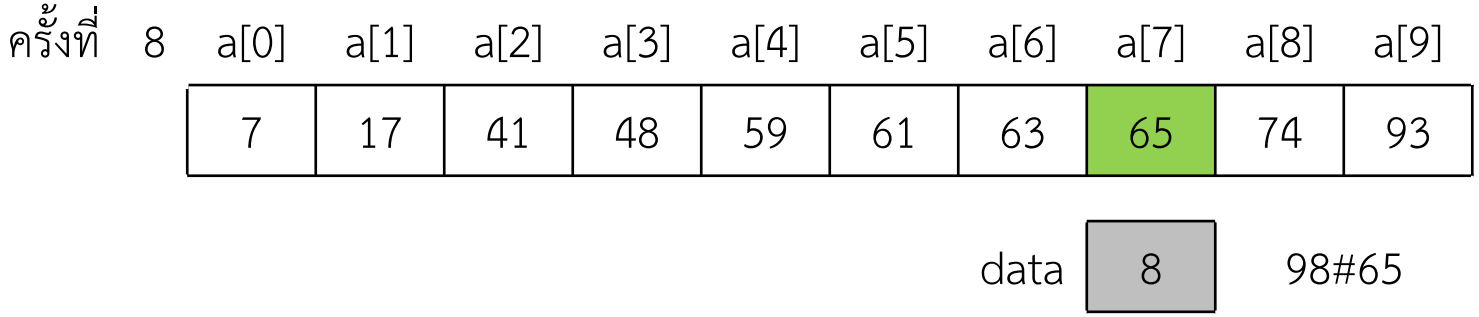
- เพื่อค้นหาค่า 98 กรณีไม่พบข้อมูลที่ต้องการ



10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.3 แสดงขั้นตอนการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ(ต่อ)

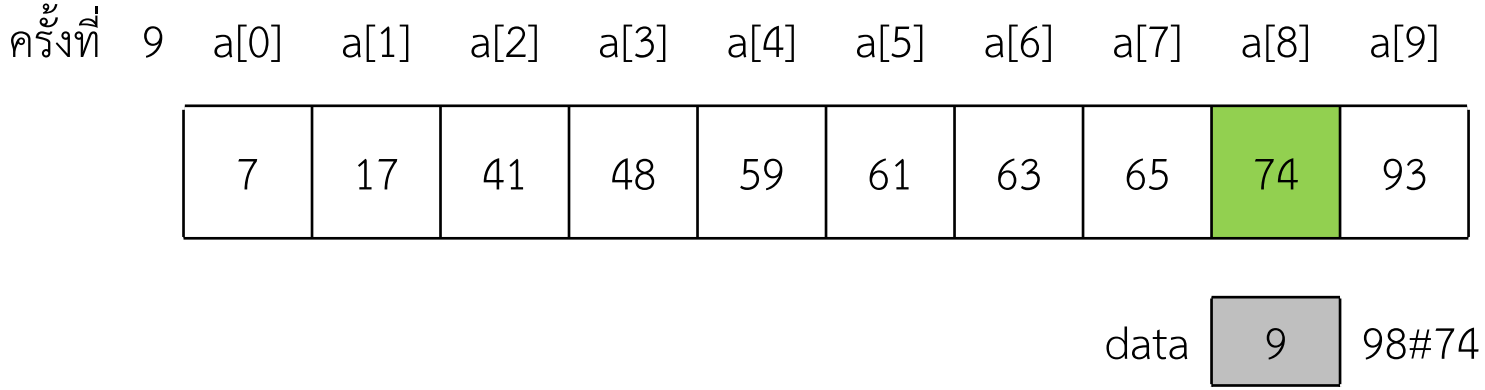
- เพื่อค้นหาค่า 98 กรณีไม่พบข้อมูลที่ต้องการ



10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.3 แสดงขั้นตอนการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ(ต่อ)

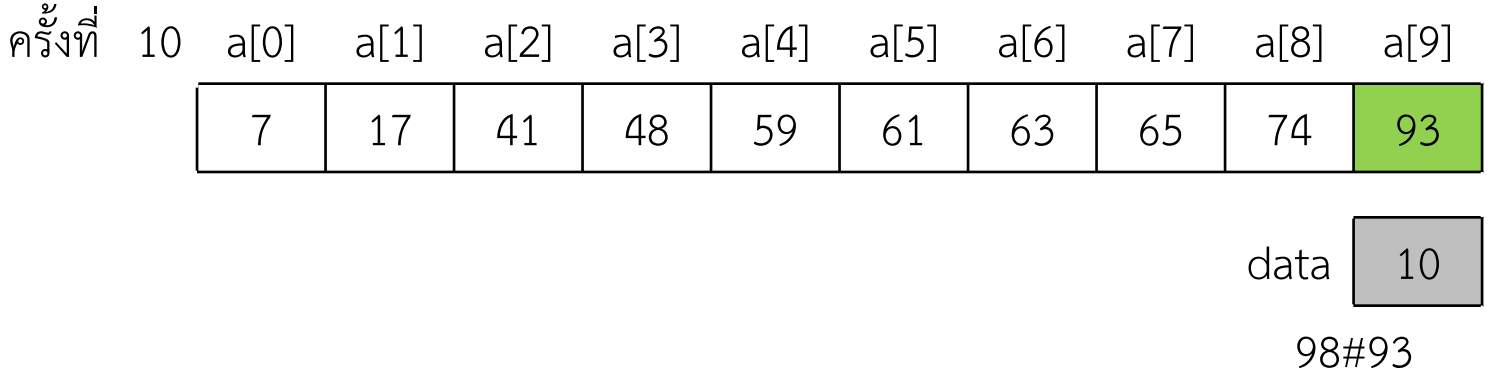
- เพื่อค้นหาค่า 98 กรณีไม่พบข้อมูลที่ต้องการ



10.1 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

ตัวอย่างที่ 10.3 แสดงขั้นตอนการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ(ต่อ)

- เพื่อค้นหาค่า 98 กรณีไม่พบข้อมูลที่ต้องการ



10.2 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับแบบแสดงข้อมูลหลายครั้ง

- การค้นหาข้อมูลแบบเรียงลำดับนั้นปกติเมื่อค้นพบข้อมูลจะทำการแจ้งผลการค้นพบข้อมูลที่ต้องการค้นหาและออกจากการทำงาน
- แต่เนื่องจากบางครั้งชุดข้อมูลที่นำมาค้นหานั้นพบข้อมูลที่ต้องการอยู่หลายตำแหน่ง
- ดังนั้นการค้นหาข้อมูลแบบแสดงข้อมูลหลายครั้งด้วยเทคนิค sequential searching สามารถนำเสนอดังตัวอย่างดังนี้

10.2 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับแบบแสดงข้อมูลหลายครั้ง

ตัวอย่างที่ 10.4 โปรแกรมการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับแบบแสดงข้อมูลหลายครั้ง

```
1   void sequential_search_rep
    (int data[], int size, int search)
2   {
3     int  count=0;
4     for (int i=0; i < size ; i++)
5     {
6       if( search == data[i] )
7       {
8           found = 1;
9           count=count+1;
10          cout <<endl<<search <<"
            SUCCESSFUL “;
11          cout<<” SEARCH! found at location
            "<<i+1;
12      }
13    }
```

```
14      if(found==1)
15      {
16          cout <<endl<<search <<" Found “
                <<count<<" time in the list.";
17          }else {
18          cout <<"Search is FAILED "<< search
                <<" is not present in the list.\n";
19      }
20  }
```

10.2 การค้นหาข้อมูลด้วยเทคนิคเรียงลำดับแบบแสดงข้อมูลหลายครั้ง

ผลการทำงานของโปรแกรม

กรณีค้นพบข้อมูลหลายครั้ง กำหนดให้ค้นหาค่า 61 ในชุดข้อมูลที่กำหนดให้ต่อไปนี้

ชุดข้อมูล

```
int data[] = {7, 12, 41, 48, 61, 61, 61, 65, 74, 93};
```

ผลการค้นหาข้อมูลจากชุดข้อมูลที่กำหนดให้ข้างต้น

```
Data for Search: 7 12 41 48 61 61 61 65 74 93
Enter the element to be searched : 61

61 SUCCESSFUL SEARCH! found at location 5
61 SUCCESSFUL SEARCH! found at location 6
61 SUCCESSFUL SEARCH! found at location 7
61 Found 3 time in the list.
```

10.3 การวิเคราะห์ประสิทธิภาพการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

- การค้นหาด้วยวิธีเรียงลำดับนี้จะทำการเปรียบเทียบข้อมูลทีละตัวจนกว่าจะพบข้อมูลที่ต้องการ แล้วจึงหยุดการทำงาน
- แต่ถ้าการดำเนินการค้นหายังไม่พบข้อมูลที่ต้องการก็จะดำเนินการเปรียบเทียบไปจนหมดข้อมูลด้
- การทำงานค้นหาแบบเรียงลำดับนี้จะพบว่าการทำงานเหมาะกับข้อมูลที่มีจำนวนน้อย
- การวิเคราะห์ประสิทธิภาพการค้นหาของอัลกอริทึมสามารถเปรียบเทียบแยกพิจารณากรณี ดังนี้

10.3 การวิเคราะห์ประสิทธิภาพการค้นหาข้อมูลด้วยเทคนิคเรียงลำดับ

- **กรณีดีที่สุด (best case)** พบข้อมูลที่ต้องการค้นหาในตำแหน่งแรกของข้อมูล ดังนั้นประสิทธิภาพการค้นหาจะเท่ากับ O(1)
- **กรณีที่แย่ที่สุด (worst-case)** เป็นประสิทธิภาพด้านตรงข้ามกับกรณีที่ดีที่สุดนั่นคือ การเปรียบเทียบการค้นหาข้อมูลจะพบข้อมูลที่ต้องการที่ตำแหน่งสุดท้ายของโครงสร้างข้อมูล ดังนั้นประสิทธิภาพการค้นหาจะเท่ากับ O(n)
- **กรณีเฉลี่ย (average case)** ประสิทธิภาพการค้นหาข้อมูลพบในช่วงเวลาระหว่างข้อมูลทั้งหมด ดังนั้น ถ้า n คือจำนวนข้อมูลตำแหน่ง n/2 ของอาเรย์ข้อมูลจะเป็นตำแหน่งที่พบข้อมูลที่ต้องการ

ดังนั้นประสิทธิภาพการค้นหาจะเท่ากับ $\frac{1+2+3+\cdots+n}{n} = \frac{n(n+1)}{2n} = \frac{1}{2}(n+1)$

10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

ลักษณะข้อมูลที่นำมาทำการค้นหาด้วยเทคนิคไบนารี่ (binary search)

- ต้องเก็บข้อมูลแบบเรียงลำดับ
- สามารถใช้ได้กับโครงสร้างข้อมูลที่มีปริมาณมาก
- หลักการค้นหา คือ นำชุดข้อมูลมาแบ่งครึ่งเพื่อหาข้อมูลที่ต้องการว่าอยู่ในช่วงใด
  - ➢ ถ้าน้อยกว่าก็จะไม่พิจารณาส่วนหลัง
  - ➢ ถ้ามากกว่าก็ไม่พิจารณาส่วนหน้า
  - ➢ ทำการแบ่งชุดข้อมูลไปเรื่อย ๆ จนกระทั่งพบข้อมูลที่ต้องการ

10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

การทำงานของอัลกอริทึมการค้นหาข้อมูลด้วยเทคนิคไบนารี่ สามารถอธิบายได้ดังนี้

1) กำหนดข้อมูลที่ต้องการค้นหา (search)

2) กำหนดค่า index ให้กับค่า first ปกติมีค่าเท่ากับ index เริ่มต้นมีค่าเท่ากับ 0

3) กำหนดค่า index ให้กับค่า last ปกติมีค่าเท่ากับ index ตัวสุดท้ายของชุดข้อมูลที่นำมาทำการ ค้นหา

4) คำนวณหาค่า middle จากสมการ

$$\text{middle} = \frac{\text{last+first}}{2}$$

10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

5) ถ้า first มีค่าน้อยกว่า last

5.1) ให้เลือกทำงานตามเงื่อนไข ดังนี้

5.1.1) ถ้าในค่าตำแหน่ง middle น้อยกว่า ค่าที่ต้องการค้นหา (search)

กำหนดให้ตัวแปร first = middle+1

5.2.2) ถ้าในค่าตำแหน่ง middle เท่ากับ ค่าที่ต้องการค้นหา (search)

แสดงข้อความ “ข้อมูลที่ต้องการค้นหา SUCCESSFUL SEARCH! found at location ตำแหน่งที่พบ”

5.3.3) ถ้าในค่าตำแหน่ง middle มากกว่า ค่าที่ต้องการค้นหา (search)

กำหนดให้ตัวแปร last = middle – 1

5.2 กำหนดให้ตัวแปร middle = (first + last)/2

6) ถ้า first มีค่ามากกว่า last

ให้แสดงข้อความ “Search is FAILED ข้อมูล is not present in the list.”

10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

ตัวอย่างที่ 10.5 จงเขียนโปรแกรมเพื่อการค้นหาข้อมูลด้วยเทคนิคไบนารี่

```
1  #include <iostream>
2  using namespace std;
3
4  void display (int [], int);
5  void binary_search (int [], int, int);
6  void binary_search_2 (int data[], int size,
7  int search, int first, int middle, int last);
8  int size=10;
```

```
9  int main()
10 {
11 int   search ;
12 int data[] = {7, 12, 41, 48, 59, 61,
                63, 65, 74, 93};
13 cout << "Data for Search: ";
14 display (data, size);
15 cout <<"Enter the element to be searched
16 :";
17 cin>>search;
18 binary_search (data, size, search);
19 return 0;
   }
```

10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

ตัวอย่างที่ 10.5 จงเขียนโปรแกรมเพื่อการค้นหาข้อมูลด้วยเทคนิคไบนารี่

```
20  void binary_search (int data[], int size, int search)
21  {
22     int first = 0;
23     int last = size - 1;
24     int middle = (first+last)/2;
25     binary_search_2 (data, size, search, first, middle, last);
26     if (first > last)
27     {
28        cout <<"Search is FAILED ";
29        cout << search <<" is not present in the list.\n";
30     }
31  }
```

10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

ตัวอย่างที่ 10.5 จงเขียนโปรแกรมเพื่อการค้นหาข้อมูลด้วยเทคนิคไบนารี่

```
32  void binary_search_2 (int data[], int size,
33  int search, int first, int middle, int last)
34  {
35     while (first <= last)
36     {
37        if (data[middle] < search)
38        {
39           first = middle + 1;
40           }else if (data[middle] == search) {
41           cout << search <<" SUCCESSFUL SEARCH!";
42           cout <<"found at location "<<middle+1;
43           break;
44           } else {
45           last = middle - 1;
46        }
47        middle = (first + last)/2;
48     }
49  }
```

10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

ตัวอย่างที่ 10.5 จงเขียนโปรแกรมเพื่อการค้นหาข้อมูลด้วยเทคนิคไบนารี่

```
50  void display (int data[], int size)
51  {
52          for (int i = 0; i < size; ++i)
53          {
54                  cout << data[i] << " ";
55          }
```

10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

ผลการทำงานของโปรแกรม

```
Data for Search: 7 12 41 48 59 61 63 65 74 93
Enter the element to be searched :63
63 SUCCESSFUL SEARCH! found at location 7
```

10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

ตัวอย่างที่ 10.6 การค้นหาข้อมูลด้วยเทคนิค binary search

เพื่อค้นหาค่า 63 **กรณี** พบข้อมูลที่ต้องการ จากข้อมูลดังต่อไปนี้

กรณี พบข้อมูลที่ต้องการ จากข้อมูลดังต่อไปนี้

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]
7	17	41	48	59	61	63	65	74	93

10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

การทำงานของอัลกอริทึม

รอบที่ 1 search = 63

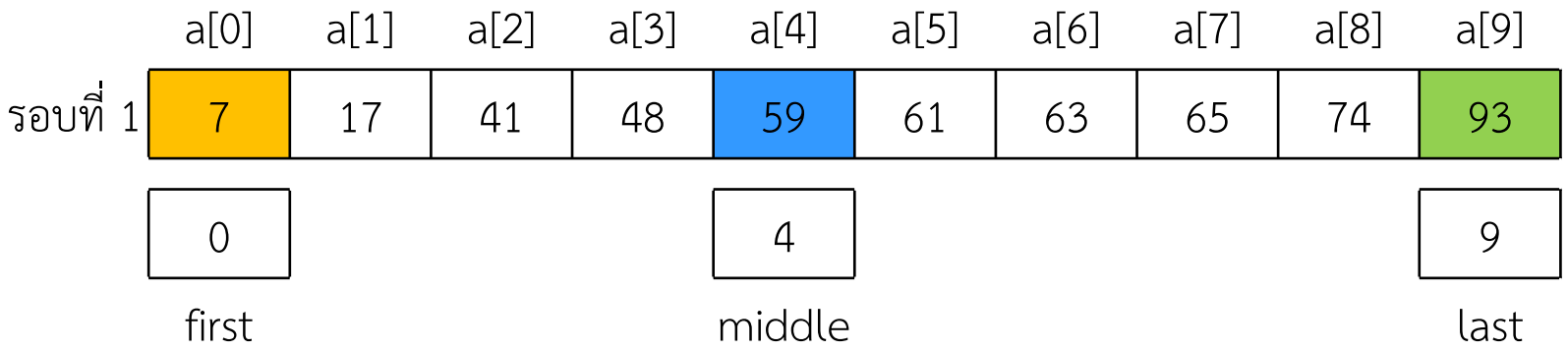
1) index ค่า first = 0

2) index ค่า last = 9

3) index ค่า middle = $\frac{0+9}{2} = 4$ (ปัดเศษทิ้ง)

อธิบายการทำงาน

•



10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

การทำงานของอัลกอริทึม

รอบที่ 2 search = 63

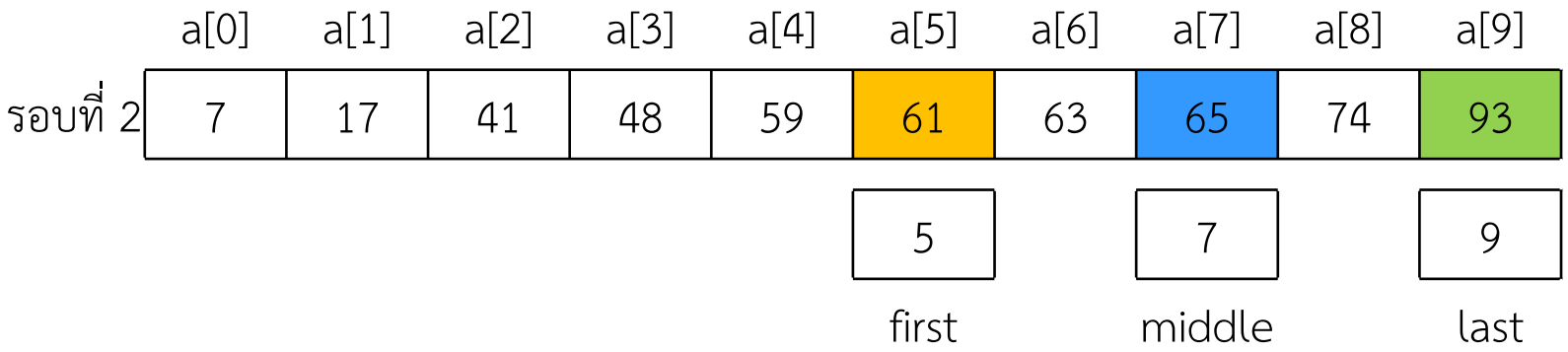
1) middle < search = 59 < 63

2) index ค่า first = 4+1=5

3) index ค่า middle = $\frac{5+9}{2} = 7$ (ปัดเศษทิ้ง)

อธิบายการทำงาน

•



10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

การทำงานของอัลกอริทึม

รอบที่ 3 search = 63

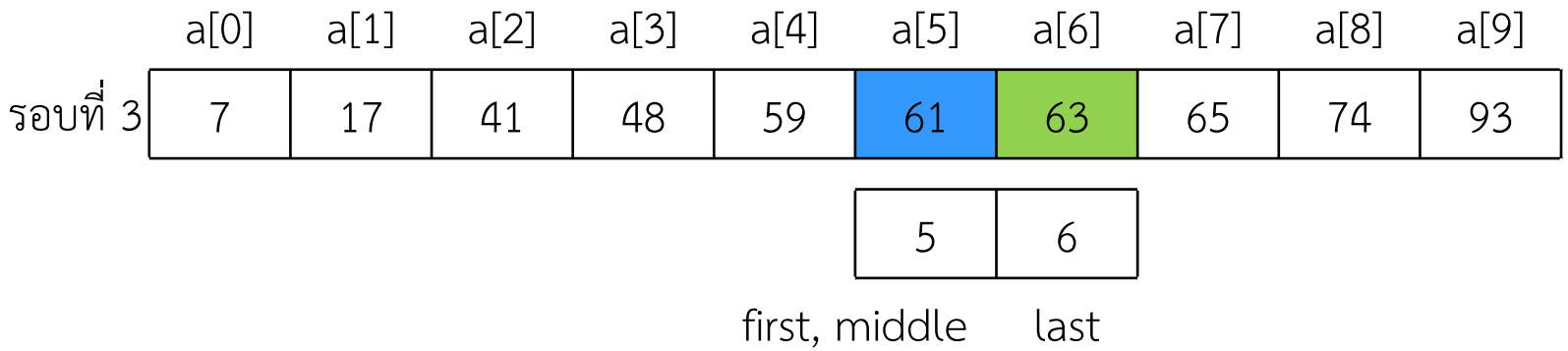
1) middle > search = 65 > 63

2) index ค่า last = 7-1=6

3) index ค่า middle = $\frac{5+6}{2} = 5$ (ปัดเศษทิ้ง)

อธิบายการทำงาน

•



10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

การทำงานของอัลกอริทึม

รอบที่ 4 search = 63

1) middle < search = 61 < 63

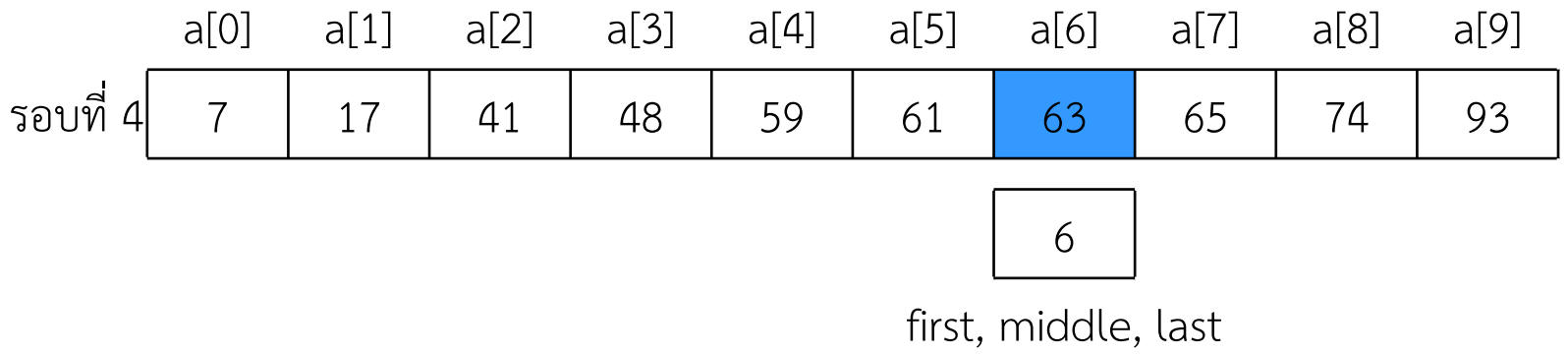
2) index ค่า first = 5+1=6

3) index ค่า middle = $\frac{6+6}{2} = 6$ (ปัดเศษทิ้ง)

แสดงข้อความเมื่อพบข้อมูลทีทำการค้นหา “63 SUCCESSFUL SEARCH! found at location 7”

อธิบายการทำงาน

•



10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

ตัวอย่างที่ 10.7 การค้นหาข้อมูลด้วยเทคนิค binary search

เพื่อค้นหาค่า 80

กรณี ไม่พบข้อมูลที่ต้องการ จากข้อมูลดังต่อไปนี้

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]
7	17	41	48	59	61	63	65	74	93

10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

การทำงานของอัลกอริทึม

รอบที่ 1 search = 80

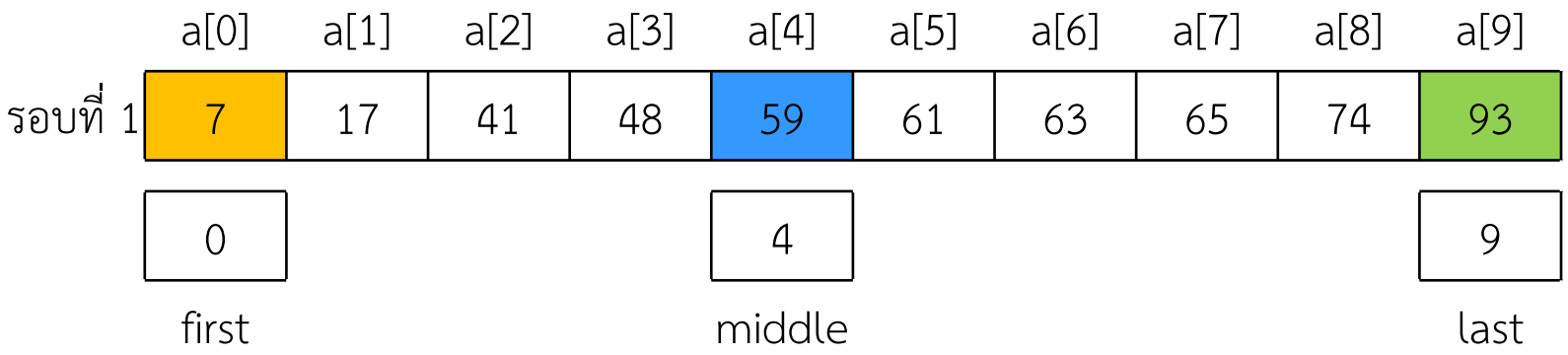
1) index ค่า first = 0

2) index ค่า last = 9

3) index ค่า middle = $\frac{0+9}{2} = 4$ (ปัดเศษทิ้ง)

อธิบายการทำงาน

•



10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

การทำงานของอัลกอริทึม

รอบที่ 2 search = 80

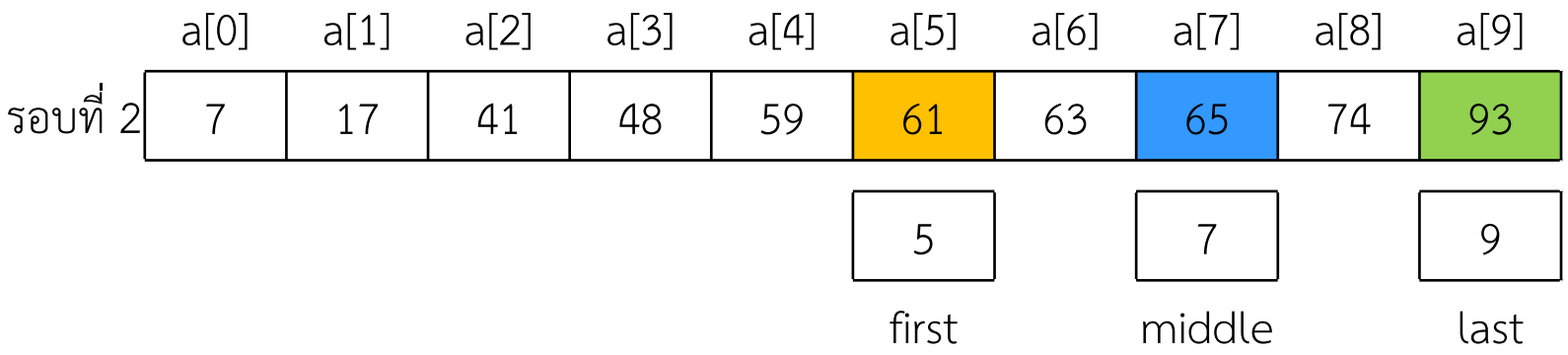
1) middle < search = 59 < 80

2) index ค่า first = 4+1=5

3) index ค่า middle = $\frac{5+9}{2} = 7$ (ปัดเศษทิ้ง)

อธิบายการทำงาน

•



10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

การทำงานของอัลกอริทึม

รอบที่ 3 search = 80

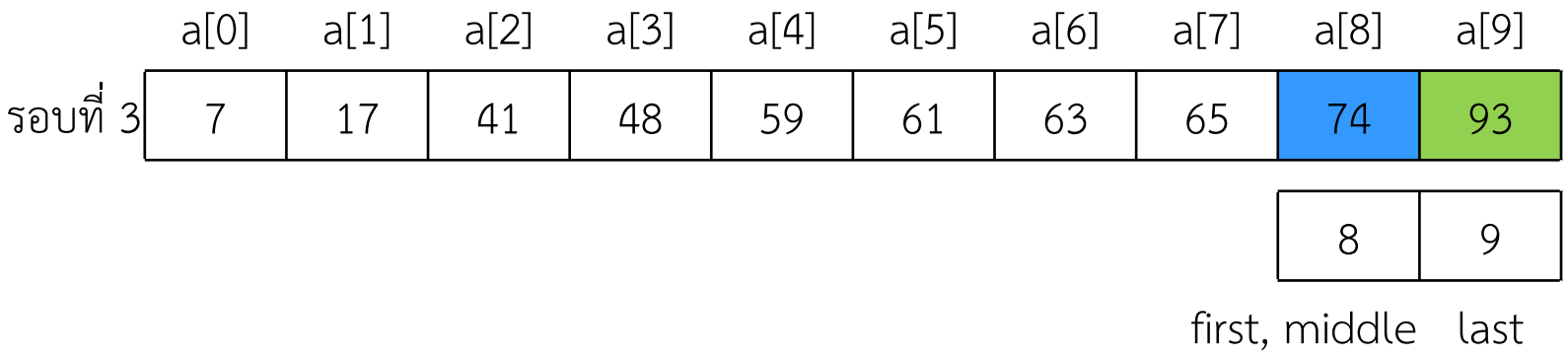
1) middle < search = 65 < 80

2) index ค่า first = 7+1=8

3) index ค่า middle = $\frac{8+9}{2} = 8$ (ปัดเศษทิ้ง)

อธิบายการทำงาน

•



10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

การทำงานของอัลกอริทึม

รอบที่ 4 search = 80

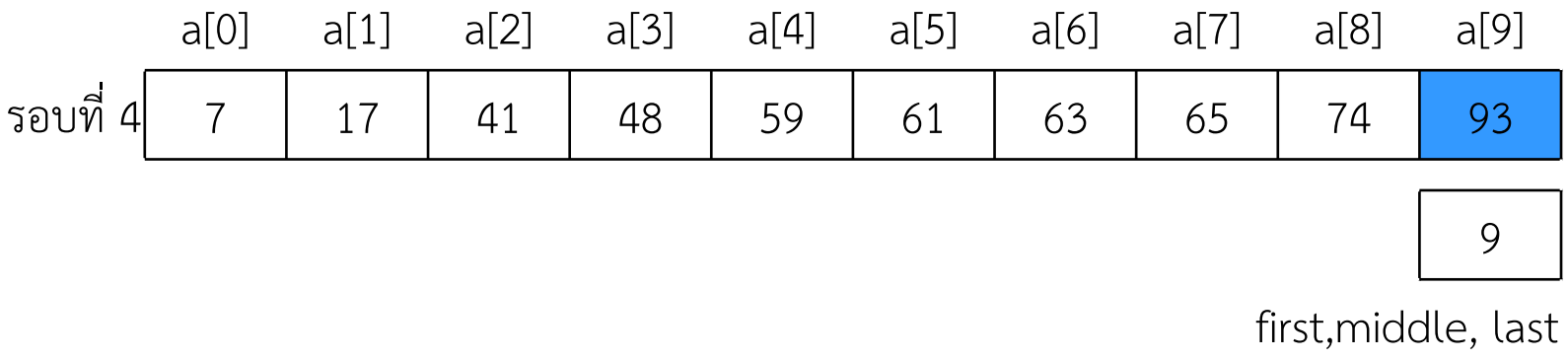
1) middle < search = 74 < 80

2) index ค่า first = 8+1=9

3) index ค่า middle = $\frac{9+9}{2} = 9$ (ปัดเศษทิ้ง)

อธิบายการทำงาน

•



10.4 การค้นหาข้อมูลด้วยเทคนิคไบนารี่

การทำงานของอัลกอริทึม

รอบที่ 5 search = 80

1) middle > search = 93 > 80

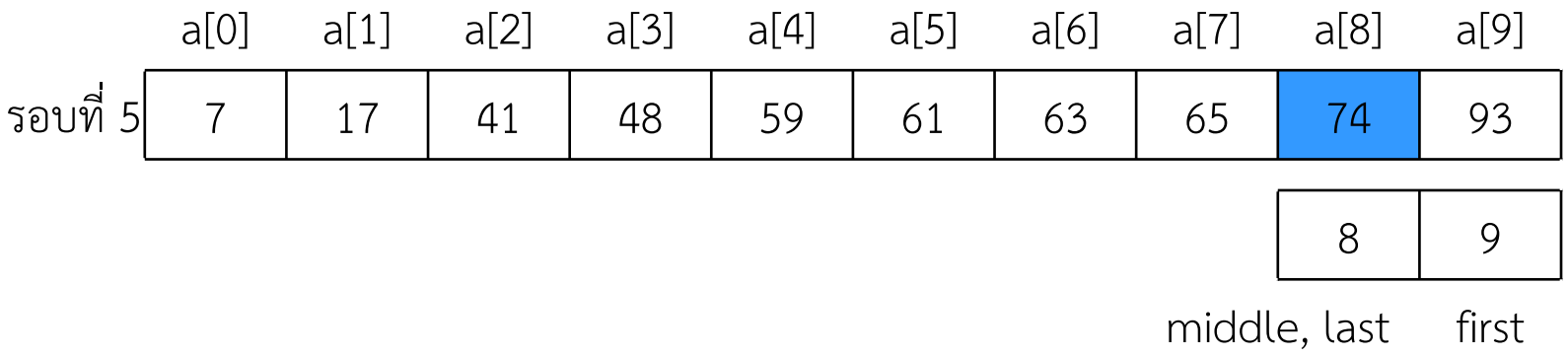
2) index ค่า last = 9-1=8

3) index ค่า middle = $\frac{9+8}{2} = 8$ (ปัดเศษทิ้ง)

เนื่องจาก ตัวแปร first มากกว่า ตัวแปร last
ดังนั้น แสดงข้อความเมื่อไม่พบข้อมูลทีทำการค้นหา
“Search is FAILED 80 is not present in the list.”

อธิบายการทำงาน

•



10.5 การวิเคราะห์ประสิทธิภาพการทำงานแบบไบนารี

ตัวอย่างที่ 10.8 การวิเคราะห์ประสิทธิภาพการค้นหาข้อมูลแบบไบนารี

การวิเคราะห์ประสิทธิภาพการค้นหาข้อมูลแบบไบนารี จะมีประสิทธิภาพการค้นหาที่ดีกว่าการค้นหาแบบอื่น ๆ ยิ่งถ้ามีข้อมูลยิ่งมาก ประสิทธิภาพจะดีกว่าวิธีการค้นหาวิธีอื่นอย่างชัดเจน
ดังนั้นประสิทธิภาพการค้นหาข้อมูลแบบไบนารีจะเท่ากับ $O(log_2 n)$

ขนาดข้อมูล	ไบนารี่	เรียงลำดับ	
		กรณีเฉลี่ย	กรณีแย่ที่สุด
16	4	8	16
50	6	25	50
256	8	128	256
1000	10	500	1000
10000	14	5000	10000
100000	17	50000	100000
1000000	20	500000	1000000

มหาวิทยาลัยราชภัฏนครปฐม