元智大學 資訊管理學系

IM112 計算機概論

期末課程輔導 – 期末考試重點整理

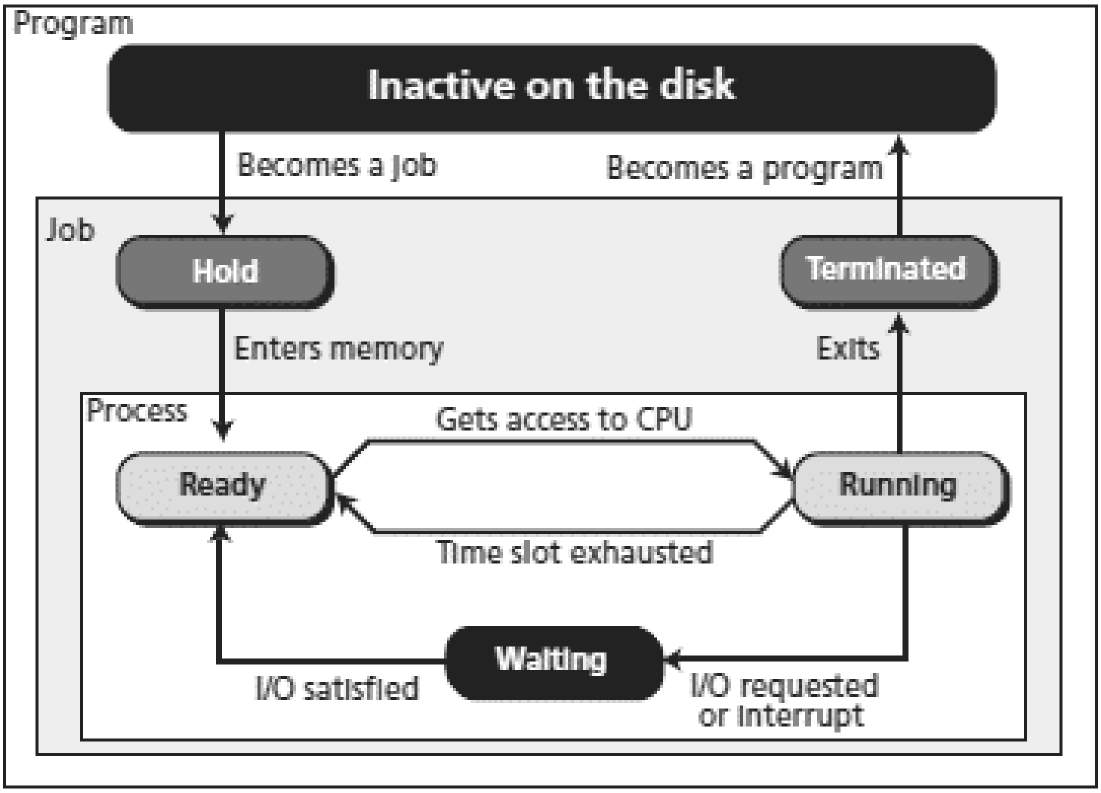
課程助教共同編著

2022/12/26

CHAPTER 7
Operating Systems

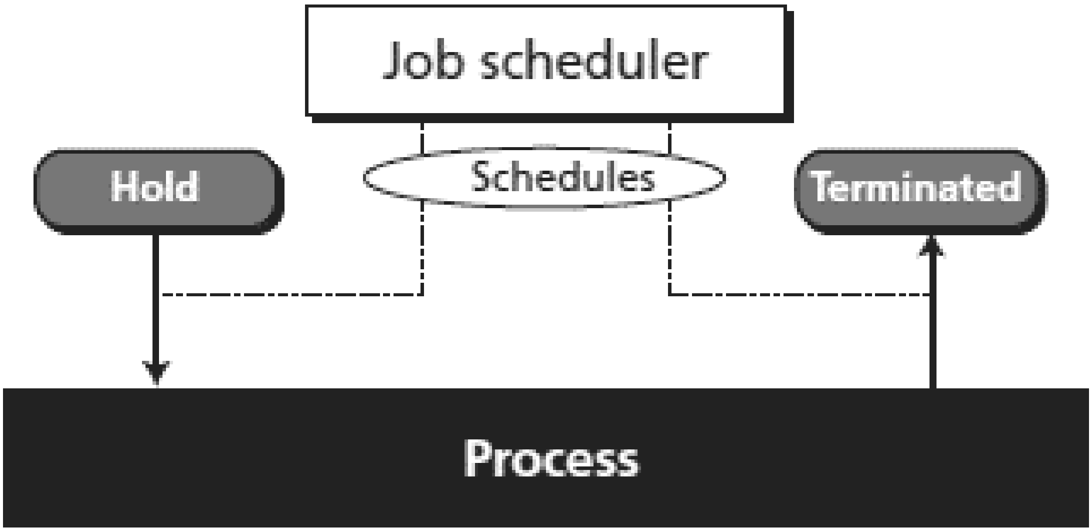
P.197

Figure 7.12 **State diagram with boundaries between program, job, and process**



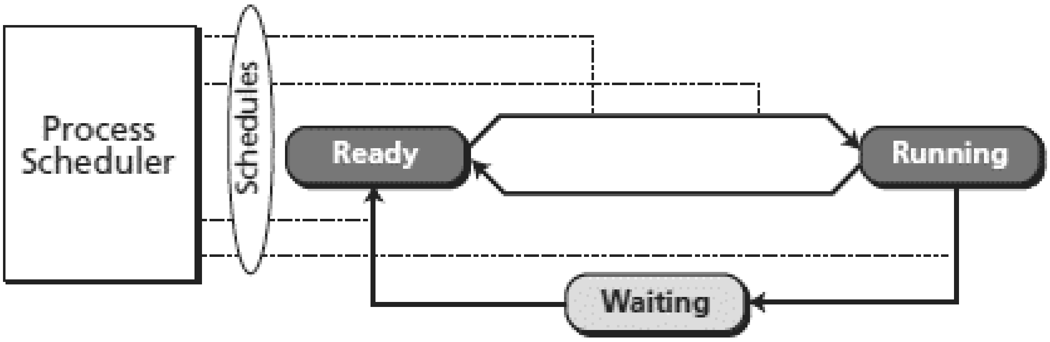
P.198

Figure 7.13 **Job scheduler**



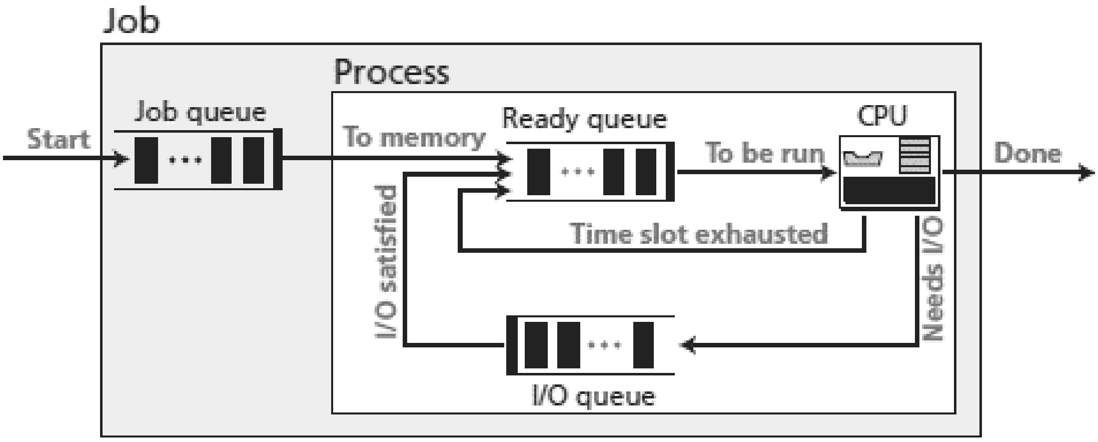
P.198

Figure 7.14 **Process scheduler**



P.199

Figure 7.15 **Queues for process management**



CHAPTER 8
Algorithms

I. Bubble Sort

i. Sorting from left to right in ascending order

```
Function bubbleSort(Type data[1...n])
    Index i, j
    For i from 1 to n - 1 do
        For j from 1 to n - i do
            If data[j] > data[j + 1] then
                Exchange data[j] and data[j + 1]
End
```

ii. Sorting from left to right in descending order

```
Function bubbleSort(Type data[1...n])
    Index i, j
    For i from 1 to n - 1 do
        For j from 1 to n - i do
            If data[j] < data[j + 1] then
                Exchange data[j] and data[j + 1]
End
```

iii. Sorting from right to left in ascending order

```
Function bubbleSort(Type data[1...n])
    Index i, j
    For i from 1 to n - 1 do
        For j from n to i + 1 do
            If data[j - 1] > data[j] then
                Exchange data[j] and data[j - 1]
End
```

iv. Sorting from right to left in descending order

```
Function bubbleSort(Type data[1...n])
    Index i, j
    For i from 1 to n - 1 do
        For j from n to i + 1 do
            If data[j - 1] < data[j] then
                Exchange data[j] and data[j - 1]
End
```

II. Selection Sort

i. Sorting from left to right in ascending order

```
Function selectionSort(Type data[1...n])
    Index i, j, min
    For i from 1 to n - 1 do
        min = i
        For j from i + 1 to n do
            If data[j] < data[min] then
                min = j

        Exchange data[i] and data[min]
End
```

ii. Sorting from left to right in descending order

```
Function selectionSort(Type data[1...n])
    Index i, j, max
    For i from 1 to n - 1 do
        max = i
        For j from i + 1 to n do
            If data[j] > data[max] then
                max = j

        Exchange data[i] and data[max]
End
```

iii. Sorting from right to left in ascending order

```
Function selectionSort(Type data[1...n])
    Index i, j, max
    For i from n to 2 do
        max = i
        For j from i - 1 to 1 do
            If data[j] > data[max] then
                max = j

        Exchange data[i] and data[max]
End
```

iv. Sorting from right to left in descending order

```
Function selectionSort(Type data[1...n])
    Index i, j, min
    For i from n to 2 do
        min = i
        For j from i - 1 to 1 do
            If data[j] < data[min] then
                min = j

        Exchange data[i] and data[min]
End
```

III. Insertion Sort

i. Sorting from left to right in ascending order

```
Function insertionSort(Type data[1...n])
    Index i, j
    Type value
    For i from 2 to n do
        value = data[i]

        j = i - 1
        While j >= 1 and data[j] > value do
            data[j + 1] = data[j]
            j = j - 1

        data[j + 1] = value
End
```

ii. Sorting from left to right in descending order

```
Function insertionSort(Type data[1...n])
    Index i, j
    Type value
    For i from 2 to n do
        value = data[i]

        j = i - 1
        While j >= 1 and data[j] < value do
            data[j + 1] = data[j]
            j = j - 1

        data[j + 1] = value
End
```

iii. Sorting from right to left in ascending order

```
Function insertionSort(Type data[1...n])
    Index i, j
    Type value
    For i from n - 1 to 1 do
        value = data[i]

        j = i + 1
        While j <= n and data[j] < value do
            data[j - 1] = data[j]
            j = j + 1

        data[j - 1] = value
End
```

iv. Sorting from right to left in descending order

```
Function insertionSort(Type data[1...n])
    Index i, j
    Type value
    For i from n - 1 to 1 do
        value = data[i]

        j = i + 1
        While j <= n and data[j] > value do
            data[j - 1] = data[j]
            j = j + 1

        data[j - 1] = value
End
```

IV. Binary Search

i. A recursive approach

```
Function binarySearch (Type data[1...n], Type search, Index low, Index high)
    If low > high then
        return NotFound
    Else
        Index mid = (low + high) / 2
        If data[mid] = search then
            return mid
        Else if data[mid] > search then
            return binarySearch(data, search, low, mid - 1)
        Else if data[mid] < search then
            return binarySearch(data, search, mid + 1, high)
End
```

ii. An iterative approach

```
Function binarySearch (Type data[1...n], Type search)
    Index low = 1
    Index high = n

    while low <= high do
        Index mid = (low + high) / 2
        If data[mid] = search then
            return mid
        Else if data[mid] > search then
            high = mid - 1
        Else if data[mid] < search then
            low = mid + 1

    return NotFound
End
```

V. Factorial of n

i. A recursive approach

```
int Factorial (int n)
    If n = 0 then
        return 1
    Else
        return n * Factorial (n - 1)
End
```

ii. An iterative approach

```
int Factorial (int n)
    int result = 1
    int i = 1

    while i <= n do
        result = result * i
        i = i + 1

    return result
End
```

VI. Fibonacci Sequence

A recursive approach – Given base case F(0)=0, F(1)=1

```
int Fibonacci (int n)
    If n = 0
        return 0
    Else if n = 1
        return 1
    Else
        return Fibonacci (n - 1) + Fibonacci(n - 2)
End
```

VII. Linear Search

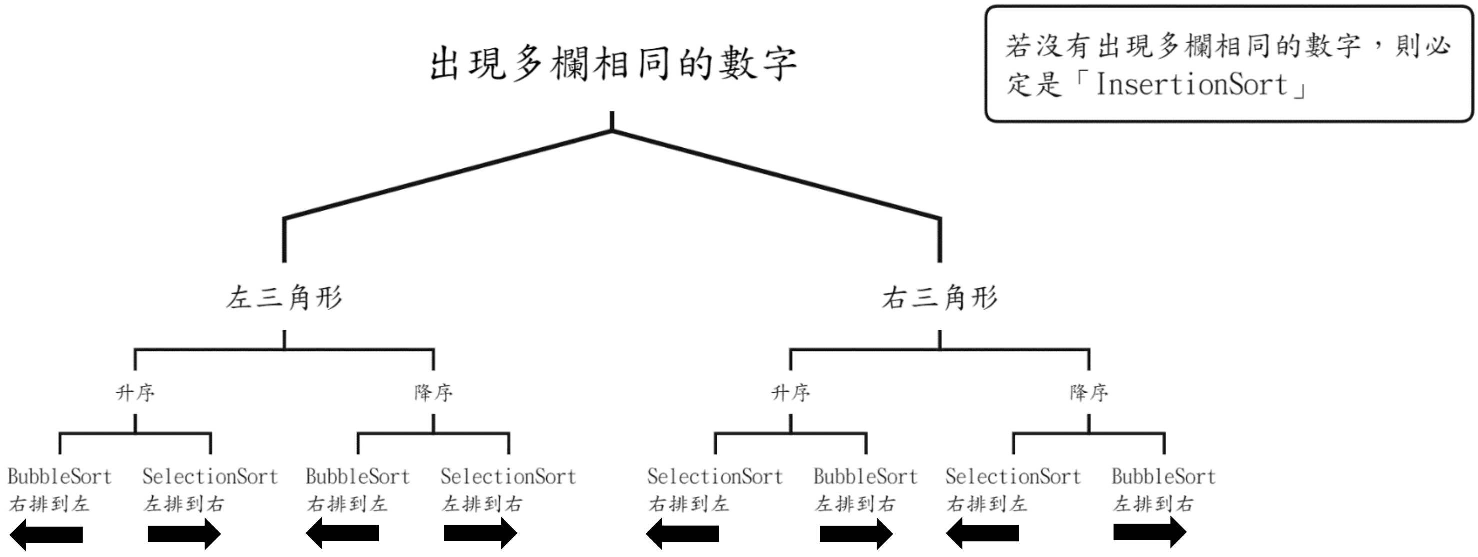
```
Function linearSearch(Type data[1…n], Type search)
    Index i

    For i from 1 to n do
        If data[i] = search
            return i

    return NotFound

End
```

VIII. How to determine the algorithm given the result?



若沒有出現多欄相同的數字，則必定是「InsertionSort」

i. Bubble Sorting process

left to right, ascending order

Original Data:3 5 2 1 8 7 6 4

Pass1: 3 2 1 5 7 6 4 **8**

Pass2: 2 1 3 5 6 4 **7 8**

Pass3: 1 2 3 5 4 **6 7 8**

Pass4: 1 2 3 4 **5 6 7 8**

Pass5: 1 2 3 **4 5 6 7 8**

Pass6: 1 2 **3 4 5 6 7 8**

Pass7: 1 **2 3 4 5 6 7 8**

Final result:1 2 3 4 5 6 7 8

right to left, ascending order

Original Data:3 5 2 1 8 7 6 4

Pass1: **1** 3 5 2 4 8 7 6

Pass2: **1 2** 3 5 4 6 8 7

Pass3: **1 2 3** 4 5 6 7 8

Pass4: **1 2 3 4** 5 6 7 8

Pass5: **1 2 3 4 5** 6 7 8

Pass6: **1 2 3 4 5 6** 7 8

Pass7: **1 2 3 4 5 6 7** 8

Final result:1 2 3 4 5 6 7 8

left to right, descending order

Original Data:3 5 2 1 8 7 6 4

Pass1: 5 3 2 8 7 6 4 **1**

Pass2: 5 3 8 7 6 4 **2 1**

Pass3: 5 8 7 6 4 **3 2 1**

Pass4: 8 7 6 5 **4 3 2 1**

Pass5: 8 7 6 **5 4 3 2 1**

Pass6: 8 7 **6 5 4 3 2 1**

Pass7: 8 **7 6 5 4 3 2 1**

Final result:8 7 6 5 4 3 2 1

right to left, descending order

Original Data:3 5 2 1 8 7 6 4

Pass1: **8** 3 5 2 1 7 6 4

Pass2: **8 7** 3 5 2 1 6 4

Pass3: **8 7 6** 3 5 2 1 4

Pass4: **8 7 6 5** 3 4 2 1

Pass5: **8 7 6 5 4** 3 2 1

Pass6: **8 7 6 5 4 3** 2 1

Pass7: **8 7 6 5 4 3 2** 1

Final result:8 7 6 5 4 3 2 1

ii. Selection Sorting process

left to right, ascending order

Original Data:3 5 2 1 8 7 6 4

Pass1: **1** 5 2 3 8 7 6 4

Pass2: **1 2** 5 3 8 7 6 4

Pass3: **1 2 3** 5 8 7 6 4

Pass4: **1 2 3 4** 8 7 6 5

Pass5: **1 2 3 4 5** 7 6 8

Pass6: **1 2 3 4 5 6** 7 8

Pass7: **1 2 3 4 5 6 7** 8

Final result:1 2 3 4 5 6 7 8

left to right, descending order

Original Data:3 5 2 1 8 7 6 4

Pass1: **8** 5 2 1 3 7 6 4

Pass2: **8 7** 2 1 3 5 6 4

Pass3: **8 7 6** 1 3 5 2 4

Pass4: **8 7 6 5** 3 1 2 4

Pass5: **8 7 6 5 4** 1 2 3

Pass6: **8 7 6 5 4 3** 2 1

Pass7: **8 7 6 5 4 3 2** 1

Final result:8 7 6 5 4 3 2 1

right to left, ascending order

Original Data:3 5 2 1 8 7 6 4

Pass1: 3 5 2 1 4 7 6 **8**

Pass2: 3 5 2 1 4 6 **7 8**

Pass3: 3 5 2 1 4 **6 7 8**

Pass4: 3 4 2 1 **5 6 7 8**

Pass5: 3 1 2 **4 5 6 7 8**

Pass6: 2 1 **3 4 5 6 7 8**

Pass7: 1 **2 3 4 5 6 7 8**

Final result:1 2 3 4 5 6 7 8

right to left, descending order

Original Data:3 5 2 1 8 7 6 4

Pass1: 3 5 2 4 8 7 6 **1**

Pass2: 3 5 6 4 8 7 **2 1**

Pass3: 7 5 6 4 8 **3 2 1**

Pass4: 7 5 6 8 **4 3 2 1**

Pass5: 7 8 6 **5 4 3 2 1**

Pass6: 7 8 **6 5 4 3 2 1**

Pass7: 8 **7 6 5 4 3 2 1**

Final result:8 7 6 5 4 3 2 1

iii. Insertion Sorting process

left to right, ascending order

Original Data:3 5 2 1 8 7 6 4

Pass1: 3 5 2 1 8 7 6 4

Pass2: 2 3 5 1 8 7 6 4

Pass3: 1 2 3 5 8 7 6 4

Pass4: 1 2 3 5 8 7 6 4

Pass5: 1 2 3 5 7 8 6 4

Pass6: 1 2 3 5 6 7 8 4

Pass7: 1 2 3 4 5 6 7 8

Final result:1 2 3 4 5 6 7 8

right to left, ascending order

Original Data:3 5 2 1 8 7 6 4

Pass1: 3 5 2 1 8 7 4 6

Pass2: 3 5 2 1 8 4 6 7

Pass3: 3 5 2 1 4 6 7 8

Pass4: 3 5 2 1 4 6 7 8

Pass5: 3 5 1 2 4 6 7 8

Pass6: 3 1 2 4 5 6 7 8

Pass7: 1 2 3 4 5 6 7 8

Final result:1 2 3 4 5 6 7 8

left to right, descending order

Original Data:3 5 2 1 8 7 6 4

Pass1: 5 3 2 1 8 7 6 4

Pass2: 5 3 2 1 8 7 6 4

Pass3: 5 3 2 1 8 7 6 4

Pass4: 8 5 3 2 1 7 6 4

Pass5: 8 7 5 3 2 1 6 4

Pass6: 8 7 6 5 3 2 1 4

Pass7: 8 7 6 5 4 3 2 1

Final result:8 7 6 5 4 3 2 1

right to left, descending order

Original Data:3 5 2 1 8 7 6 4

Pass1: 3 5 2 1 8 7 6 4

Pass2: 3 5 2 1 8 7 6 4

Pass3: 3 5 2 1 8 7 6 4

Pass4: 3 5 2 8 7 6 4 1

Pass5: 3 5 8 7 6 4 2 1

Pass6: 3 8 7 6 5 4 2 1

Pass7: 8 7 6 5 4 3 2 1

Final result:8 7 6 5 4 3 2 1

The source code is available on my GitHub repository:
https://github.com/zheshenguo/IM112-Basic-Computer-Concepts

IX. Practice questions

1. Fib(n) is a Fibonacci sequence. Please write a pseudocode using recursion to calculate Fib(n) if Fib(1)=2, Fib(2)=2 and n=1……N.
2. Please write a pseudocode (bubble sort) to arrange a number list in ascending order. It should be noted that the comparisons between numbers are from the last element of the number list, data[1…n].
3. Write a pseudocode based on the following sort: data[1…n].

   6 3 1 7

   6 3 7 1 pass 1
   6 7 3 1 pass 2
   7 6 3 1 pass 3

4. Write a pseudo code to conduct binary search using iterative method.
5. Based on the following sort, please fill in the answers in the following pseudo codes.

   3 5 4 2 1

   1 3 5 4 2 pass 1
   1 2 3 5 4 pass 2
   1 2 3 4 5 pass 3
   1 2 3 4 5 pass 4

```
Function Sort(Type data[1...n])
    Index i, j
    For i from 1 to n - 1 do
        For j from         to         do
            If (                        ) then
                (                        )

End
```

6. Based on the following sort, please fill in the answers in the following pseudo codes.

5 4 2 7 6

5 4 2 6 7 loop 1
5 4 2 6 7 loop 2
2 4 5 6 7 loop 3
2 4 5 6 7 loop 4

```
Function Sort(Type data[1...n])
    Index i, j, max
    For i from n to 2 do
        max = (        )
        For j from  (        ) to (        ) do
            If (                    ) then
                (                    )

        Exchange (                    )
End
```

7. Please sort the following number list based on the following pseudo codes.

```
Function insertionSort(Type data[1...n])
    Index i, j
    Type value
    For i from n - 1 to 1 do
        value = data[i]

        j = i + 1
        While j <= n and data[j] > value do
            data[j - 1] = data[j]
            j = j + 1

        data[j - 1] = value
End
```

Number list = [2 4 5 3 1]

8. Illustrate and explain the life cycle of a process in English.