EECS 204002
Data Structures 資料結構
Prof. REN-SONG TSAY 蔡仁松 教授
NTHU

CH. 1 BASIC CONCEPTS

1.5 Algorithm

1.5

What is an Algorithm?

An ***algorithm*** is a finite set of instructions that accomplishes a particular task (problem) and satisfies the following criteria:

- Input
  - Zero/more quantities are externally supplied.
- Output
  - At least one quantity is produced.
- Definiteness
  - Each instruction is clear and unambiguous.
- Finiteness
  - Terminate after a finite number of steps.
- Effectiveness:
  - Every instruction must be basic and easy to be computed.

1.5.1

Representation of Algorithms

- Natural languages
  - English, …etc.
- Graphic representation
  - Flowchart.
  - Feasible only if the algorithm is small and simple.
- Programming language
  - C++
  - Concise and effective!

4

1.5.1

Example: Binary Search

Problem statement: Assume we have $n \geq 1$ distinct integers that are ***sorted*** in array $A[0] \ldots A[n-1]$. Determine the existence of an integer x. If $x = A[j]$, return index j; otherwise return -1.

	A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]
A	1	3	5	8	9	17	32	50

Eg. For x=9, return index 4;
For x=10, return -1.

5

1.5.1

BS in Plain English

1. Let *left* and *right* denote the left and right ends of the list with initial value 0 and n-1.
2. Let *middle* = *(left+right)* / 2 be the middle position in the list
3. Compare A[*middle*] with x and obtain three results:
   a. x < A[*middle*]: x must be somewhere between 0 and *middle-1*. We set *right* to *middle-1*
   b. x == A[*middle*]: We return *middle*
   c. x > A[*middle*]: x must be somewhere between *middle+1* and *n-1*. We set *left* to *middle+1*.
4. If x is not found and there are still integers to check, we recalculate *middle* and repeat the above comparison.

6

1.5.1

BS in Pseudo C++ Code

```
int BinarySearch(int *A, const int x, const int n)
{ int left=0, right=n-1;

  while (left <= right)
  { // more integers to check
    int middle = (left+right)/2;
    if (x < A[middle])  right = middle-1;
    else if (x > A[middle])  left = middle+1;
    else return middle;
  } // end of while
  return -1; // not found
}
```

7

1.5.2

Recursive Algorithm

- A powerful mechanism to make your algorithm or code more clear.
- Direct recursion :
  - Function calls itself directly.
  - E.g. funcA ⇒ funcA.
- Indirect recursion:
  - Function A calls other function B that invoke the function A itself.
  - E.g. funcA ⇒ funcB ⇒ funcA.

8

A Recursively Defined Problem

The binomial coefficient

$$C(n,m) = \frac{n!}{m!(n-m)!}$$

can be computed by the recursive formula:

$$C(n,m) = C(n-1,m) + C(n-1,m-1)$$

where $C(0,0) = C(n,n) = 1$

9

Principles for Feasible Recursive Algorithms

- Termination conditions:
  - The function should return a value or stop calling itself under certain conditions.
- Decreased Parameters
  - So that each call is one step closer to a termination condition.

10

There is a "While" statement

- Replace with if-else and recursion
- In Binary Search problem…

```
int BinarySearch(int *A, const int x, const int n)
{ int left=0, right=n-1;

  while (left <= right)
  {
    …
  }
  return -1;
}
```

11

Recursive Binary Search

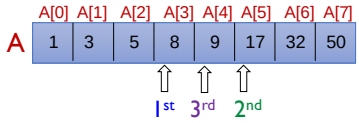
```
int BinarySearch(int *A, const int x, const int
                 left, const int right )
{ // Search the A[left],..,A[right] for x
  if (left <= right) { // more integers to check
    int middle = (left+right)/2;
    if (x < A[middle])
       return BinarySearch(A, x, left, middle-1);
    else if (x > A[middle])
       return BinarySearch(A, x, middle+1, right);
    return middle;
  } // end of if
  return -1; // not found
}
```

12

1.5.2

Example

- Search for x=9 in array A[0]…[7] :



- 1st call: `BinarySearch(A, 9, 0, 7)`
  2nd call: `BinarySearch(A, 9, 4, 7)`
  3rd call: `BinarySearch(A, 9, 4, 4)`
  return index 4.

13

14

1.5.2

Criteria of a "Good" Program

- Does it do what you want to do?
- Does it work correctly?
- Any documentation about how to use it?
- Are functions created logically?
- Is the code readable?
- However, the above questions are **HARD** to achieve (at least when only DS is taught).
- So, we focus on the "**Performance**" of the program

16