VIT UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)
VELLORE ▪ CHENNAI
www.vit.ac.in

Data Structures and Algorithms CSE220

Prof. Ramesh Ragala

July 22, 2015

Introduction to Algorithms

In general, An algorithm is a finite set of well-defined instructions that, if followed, accomplish a particular task.
In addition to this definition, all algorithms must follow five basic criteria or rules.

Introduction to Algorithms

In general, An algorithm is a finite set of well-defined instructions that, if followed, accomplish a particular task.
In addition to this definition, all algorithms must follow five basic criteria or rules.

- Input :

In general, An algorithm is a finite set of well-defined instructions that, if followed, accomplish a particular task.
In addition to this definition, all algorithms must follow five basic criteria or rules.

- Input :
  - Zero or more quantities are externally supplied to the algorithm and the logic of that algorithm processes those quantities, are called inputs.

In general, An algorithm is a finite set of well-defined instructions that, if followed, accomplish a particular task.
In addition to this definition, all algorithms must follow five basic criteria or rules.

- Input :
  - Zero or more quantities are externally supplied to the algorithm and the logic of that algorithm processes those quantities, are called inputs.
- Output:

In general, An algorithm is a finite set of well-defined instructions that, if followed, accomplish a particular task.
In addition to this definition, all algorithms must follow five basic criteria or rules.

- Input :
  - Zero or more quantities are externally supplied to the algorithm and the logic of that algorithm processes those quantities, are called inputs.
- Output:
  - At least one quantity is produced after applying the logic on input quantities.
  - These results are called outputs.
  - Every algorithm should have at least one output.

In general, An algorithm is a finite set of well-defined instructions that, if followed, accomplish a particular task.
In addition to this definition, all algorithms must follow five basic criteria or rules.

- Input :
  - Zero or more quantities are externally supplied to the algorithm and the logic of that algorithm processes those quantities, are called inputs.
- Output:
  - At least one quantity is produced after applying the logic on input quantities.
  - These results are called outputs.
  - Every algorithm should have at least one output.
- Definiteness :

In general, An algorithm is a finite set of well-defined instructions that, if followed, accomplish a particular task.
In addition to this definition, all algorithms must follow five basic criteria or rules.

- Input :
  - Zero or more quantities are externally supplied to the algorithm and the logic of that algorithm processes those quantities, are called inputs.
- Output:
  - At least one quantity is produced after applying the logic on input quantities.
  - These results are called outputs.
  - Every algorithm should have at least one output.
- Definiteness :
  - Every instruction is clear and unambiguous.
  - The steps or instructions, that are used to describe algorithm or logic of algorithm should clearly specifies, what it going to do with out any confusion.

In general, An algorithm is a finite set of well-defined instructions that, if followed, accomplish a particular task.
In addition to this definition, all algorithms must follow five basic criteria or rules.

- Input :
  - Zero or more quantities are externally supplied to the algorithm and the logic of that algorithm processes those quantities, are called inputs.
- Output:
  - At least one quantity is produced after applying the logic on input quantities.
  - These results are called outputs.
  - Every algorithm should have at least one output.
- Definiteness :
  - Every instruction is clear and unambiguous.
  - The steps or instructions, that are used to describe algorithm or logic of algorithm should clearly specifies, what it going to do with out any confusion.
  - To Achieve this criteria, algorithms are written in programming languages

- Finiteness:

- Finiteness:
  - If we trace out the instructions of an algorithm, thus for all cases, the algorithm terminates after a finite amount of time.
  - i.e, Problem can be solved by algorithmically in a finite amount of time.
  - i.e algorithm can be terminated after finite number of steps.

- Finiteness:
  - If we trace out the instructions of an algorithm, thus for all cases, the algorithm terminates after a finite amount of time.
  - i.e, Problem can be solved by algorithmically in a finite amount of time.
  - i.e algorithm can be terminated after finite number of steps.
- Effectiveness:

- Finiteness:
  - If we trace out the instructions of an algorithm, thus for all cases, the algorithm terminates after a finite amount of time.
  - i.e, Problem can be solved by algorithmically in a finite amount of time.
  - i.e algorithm can be terminated after finite number of steps.
- Effectiveness:
  - Every instruction must be very basic, so that it can be carried out, in principle, by a person using only pen and paper.
  - i.e each step of algorithm is simple, easy to understand, and also perform the processing in easy ways by anyone.
  - The operations of algorithm are not only definiteness, but also feasible.

- Finiteness:
  - If we trace out the instructions of an algorithm, thus for all cases, the algorithm terminates after a finite amount of time.
  - i.e, Problem can be solved by algorithmically in a finite amount of time.
  - i.e algorithm can be terminated after finite number of steps.
- Effectiveness:
  - Every instruction must be very basic, so that it can be carried out, in principle, by a person using only pen and paper.
  - i.e each step of algorithm is simple, easy to understand, and also perform the processing in easy ways by anyone.
  - The operations of algorithm are not only definiteness, but also feasible.

Computational Procedure: satisfies definiteness and Effectiveness

Example: Operating System of Digital Computer

Definition:

" A data structure is a systematic way of organizing and accessing the data "

it has four stages:

it has four stages:

- How to Devise an Algorithm

Life Cycle of Algorithm

it has four stages:

- How to Devise an Algorithm
  - Knowledge on problem specification and user requirement
  - choose good algorithm strategies

Life Cycle of Algorithm

it has four stages:

- How to Devise an Algorithm
  - Knowledge on problem specification and user requirement
  - choose good algorithm strategies
- How to validate an algorithm

Life Cycle of Algorithm

it has four stages:

- How to Devise an Algorithm
  - Knowledge on problem specification and user requirement
  - choose good algorithm strategies
- How to validate an algorithm
  - Algorithm Validation (first Phase)
  - Program Verification or program proving (Second Phase)

Life Cycle of Algorithm

it has four stages:

- How to Devise an Algorithm
  - Knowledge on problem specification and user requirement
  - choose good algorithm strategies
- How to validate an algorithm
  - Algorithm Validation (first Phase)
  - Program Verification or program proving (Second Phase)
- How to analyse an algorithm

Life Cycle of Algorithm

it has four stages:

- How to Devise an Algorithm
  - Knowledge on problem specification and user requirement
  - choose good algorithm strategies
- How to validate an algorithm
  - Algorithm Validation (first Phase)
  - Program Verification or program proving (Second Phase)
- How to analyse an algorithm
  - performance

Life Cycle of Algorithm

it has four stages:

- How to Devise an Algorithm
  - Knowledge on problem specification and user requirement
  - choose good algorithm strategies
- How to validate an algorithm
  - Algorithm Validation (first Phase)
  - Program Verification or program proving (Second Phase)
- How to analyse an algorithm
  - performance
    - Time Complexity
    - Space Complexity

Life Cycle of Algorithm

it has four stages:

- How to Devise an Algorithm
  - Knowledge on problem specification and user requirement
  - choose good algorithm strategies
- How to validate an algorithm
  - Algorithm Validation (first Phase)
  - Program Verification or program proving (Second Phase)
- How to analyse an algorithm
  - performance
    - Time Complexity
    - Space Complexity
- How to test a Program

Life Cycle of Algorithm

it has four stages:

- How to Devise an Algorithm
  - Knowledge on problem specification and user requirement
  - choose good algorithm strategies
- How to validate an algorithm
  - Algorithm Validation (first Phase)
  - Program Verification or program proving (Second Phase)
- How to analyse an algorithm
  - performance
    - Time Complexity
    - Space Complexity
- How to test a Program
  - debugging

Algorithm Specifications

- Distinct Difference between algorithms and programs
- The algorithm is usually described in English language to ensure definiteness condition
- Some Other ways to describe algorithms:

Algorithm Specifications

- Distinct Difference between algorithms and programs
- The algorithm is usually described in English language to ensure definiteness condition
- Some Other ways to describe algorithms:
  - **Flow Charts**
    - It is used to represent the algorithm and algorithm flow control in graphical representation.
    - This method is not efficient and makes more complex for large algorithms.
  - **Pseudo Code**
    - It is a mixture of natural language and high level programming constructs that describes the main ideas behind a generic implementation of a data structure or algorithm.
    - It is easy to read and understand
    - It should not resemble any particular programming language
    - The pseudo code is more compact than an equivalent actual software code fragment would be.

Pseudo Code Conventions

- Comments are begin with // and continue until the end of the line.
- Compound statement is represented by a block.
  Each block is indicated by matching braces only.
- Every statement is delimited by semicolon (;).
- Assigning a value to a variable done using assignment operator.
  variable := expression or variable
- It uses Boolean values (TRUE and FALSE), Logical Operators (AND, OR and NOT) and Relational Operators like $<$, $>$, $\leq$, $\geq$ and $==$.
- Elements of arrays can be accessed using subscripts braces and subscripts or indices
- READ and WRITE phases are used to specify the input and output of algorithm.

Pseudo Code Conventions

- It also uses break statement and return statement.
  - The break statement is used for force exit from loops.
  - The return statement with value is return from the specified method also exit from function it self.
- It also uses for, while and repeat-until looping statements.
- The while loop form:

```
while (condition) do
{
Statement - 1;
Statement - 2;
.
.
Statement - n;
}
```

Pseudo Code Conventions

- The for loop form:

```
for variable := value-1 to value-2 step STEP do
{
Statement - 1;
Statement - 2;
.
.
Statement - n;
}
```

- The repeat until loop form:

```
repeat {
Statement - 1;
Statement - 2;
.
.
Statement - n;
} until(condition);
```

Pseudo Code Conventions

- It also uses conditional statements like IF-THEN block, IF-THEN-ELSE block, CASE etc.
  - IF THEN block form:
    IF (condition) THEN
    Statements;
  - IF THEN ELSE block form:
    IF (condition) THEN
    Statements;
    ELSE
    Statements;
- CASE statement form:
  CASE
  {
  : condition - 1 : statement - 1;
  :condition - 2: statement - 2;
  .
  : condition - n : statement - n;
  : Else : statement - n
  }