Introduction to Data Structure

IF2030/Algoritma dan Struktur Data

Review Type Bentukan - Fungsional

- Type adalah himpunan nilai dan sekumpulan operator yang terdefinisi terhadap type tersebut
  - Dalam konteks fungsional: operator dijabarkan dalam bentuk fungsi
  - Operator: harus dibuat definisi, spesifikasi, dan realisasinya sendiri tergantung pada operasi yang berlaku pada type bentukan
- Menyatakan nilai type bentukan: dalam bentuk tuple.
  Contoh: nilai suatu Point didefinisikan oleh tuple <x:integer, y:integer>, misal: <0,0>

Mendefinisikan Type

- Dalam konteks fungsional mendefinisikan type adalah mendefinisikan:
  - Nama dan struktur type (komponen-komponennya)
  - Selektor untuk mengakses komponen-komponen type
  - Konstruktor untuk “membentuk” type
  - Predikat untuk menentukan karakteristik dan pemeriksaan besaran
  - Fungsi-fungsi lain yang didefinisikan untuk type tersebut

TYPE POINT

DEFINISI TYPE

type point: <x: real, y: real>

{<x,y> adalah sebuah point, dengan x adalah absis, y adalah ordinat}

DEFINISI DAN SPESIFIKASI SELEKTOR

Absis: point → real

{Absis(P) memberikan absis Point P}

Ordinat: point → real

{Ordinat(P) memberikan ordinat Point P}

DEFINISI DAN SPESIFIKASI KONSTRUKTOR

MakePoint: 2 real → point

{MakePoint(a,b) membentuk sebuah point dari a dan b dengan a sebagai absis dan b sebagai ordinat}

DEFINISI DAN SPESIFIKASI PREDIKAT

IsOrigin?: point → boolean

{IsOrigin? (P) benar jika P adalah titik origin yaitu titik <0,0>}

Realisasi Predikat

IsOrigin?(P): Absis(P)=0 and Ordinat(P)=0

Abstract Data Type - Prosedural

- Definisi:
  - Type
  - Primitive (operator)
    - Constructor
    - Destructor
    - Selector
    - Operator: arithmetic, relational
    - Other predicates
    - Other operations
- Spesifikasi dan body
- Contoh: Date, Time (Jam), Point, Line, Rectangle

Tingkatan Pendefinisian

- Definisi fungsional
  - Pendefinisian nama struktur data dan operator/primitif
- Representasi Lojik
  - Spesifikasi type dari struktur yang menyangkut nama type dan spesifikasi semua operator termasuk pendefinisian operator fungsional menjadi bentuk fungsi atau prosedur
- Representasi Fisik
  - Spesifikasi dari struktur data sesuai implementasinya pada memori komputer dan kesediaan bahasa pemrograman
  - Pada dasarnya, ada 2 implementasi:
    - Kontigu: elemen suatu koleksi objek berurutan posisi alamatnya dengan elemen lain
    - Berkait: posisinya dapat tidak berurutan tapi dapat ditelusuri melalui informasi berupa alamat elemen lain

Contoh ADT POINT - Bahasa Algoritmik

{* Definisi Abstract Data Type POINT ***}**
type point: <x: integer, {absis}
y: integer {ordinat}>

{* Konstruktor POINT ***}**
Function MakePoint (X: integer, y:integer) → point
{membentuk sebuah point dari x dan y dengan x sebagai absis dan y sebagai ordinat}

{* Selektor POINT ***}**
Function GetAbsis (P: point) → integer
{mengirimkan komponen absis dari P}
Function Get Ordinat (P: point) → integer
{mengirimkan komponen ordinat dari P}

{* Predikat ***}**
Function IsOrigin? (P: point) → boolean
{mengirimkan nilai benar jika P adalah titik origin yaitu titik <0,0>}

ADT Implementation in C

- One ADT in one directory:
  - Specification : header file (.h)
  - Body of primitives: .c
  - Test driver as main
- ADT specification never contains variables
- Variable declaration in the main program that uses ADT

ADT of Object Collection

- Table
- Matrix
- Linear List
- Stack
- Queue
- Tree

Table (Array)

- A consecutive set of memory location
- A set of pairs: index and value
- For each defined index, there is a value associated to the index. In mathematical term: correspondence mapping
- Static and contiguous memory allocation
- Notion of element and index
- Notion of “empty” table as a collection
- Generic table

See detail examples in lecture notes

Table (Array) Primitives

- To add (“put”) an element
- To remove/delete an element
- Traversal
- Searching
- Sorting
- Extreme value: maximum, minimum

See detail examples and solutions in lecture notes

Matrix

- Definition: two dimensional array
- Notion of row and column
- Two dimensional array vs. linear physical representation in memory:
  - By row
  - By column
- Example

See detail examples and solutions in lecture notes

Matrix Primitives

- Traversal (by columns, by rows)
- Integer matrix study:
  - Addition
  - Subtraction
  - Multiplication

  See detail examples and solutions in lecture notes

Linear List

- Definition: Linear ordered list
- Notion of:
  - First element
  - Next element: successor, predecessor
  - Address

  See detail examples and solutions in lecture notes

List Primitives

- CreateEmpty
- IsEmpty
- Insert, delete, update
- Concatenation
- Traversal
- Searching
- Find the length (number of element) of a list
- Find maximum/minimum value of list elements

List Physical Representation

- Contiguous : by table/array
- Linked representation
  - Pointer
  - Table

See detail examples and solutions in lecture notes

List Variation

- List with first and last element
- Circular list
- Double pointer list
- List with dummy element
- Possible combinations

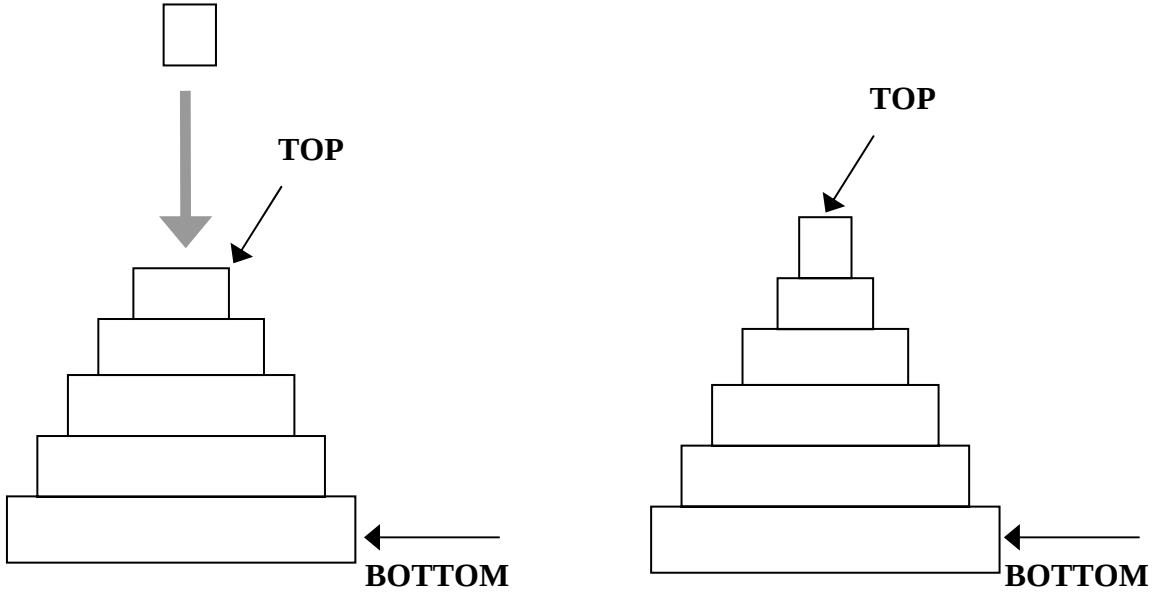
See detail examples and solutions in lecture notes

Stack

- Ordered list in which all insertions or deletions are made at one end, called the TOP
- Last in First Out (LIFO)
- Application in computer science:
  - Dynamic memory management
  - Arithmetic calculation
  - Recursive subprogram call
  - Backtracking algorithms

Stack

- Notion of:
  - TOP
  - Element
  - Empty
- Examples



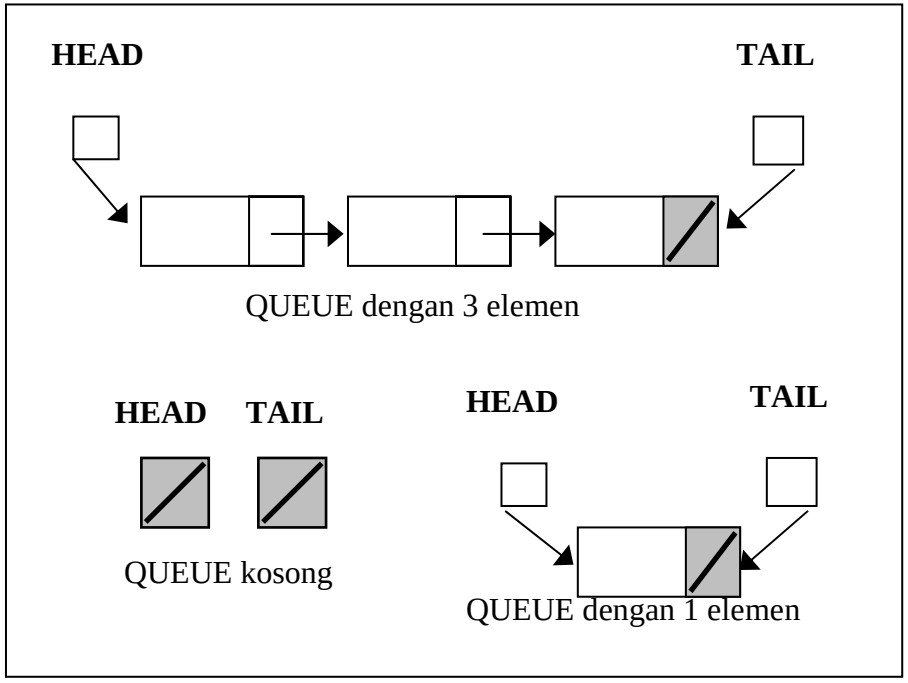
See detail examples and solutions in lecture notes

Queue

- Ordered list in which all insertions take place at one end (the rear/tail), while all deletions take place at the other end (the front/head)
- First In First Out (FIFO)
- Application of queue in computer science:
  - Job scheduling

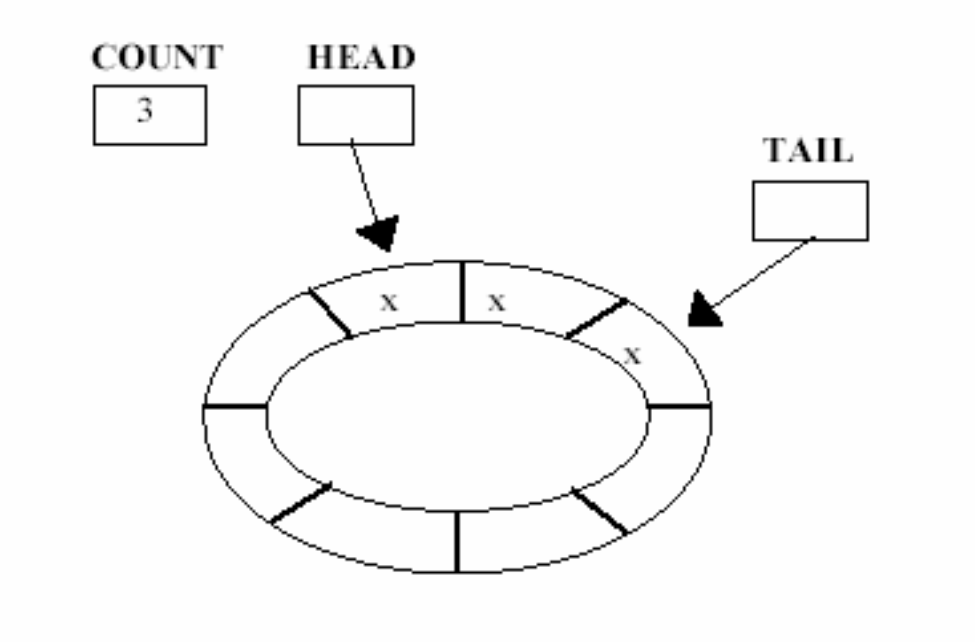
Queue

- Notion of:
  - Head
  - Tail
  - Element
  - Empty
- Examples



Queue Variation

- Circular buffered queue



Tree

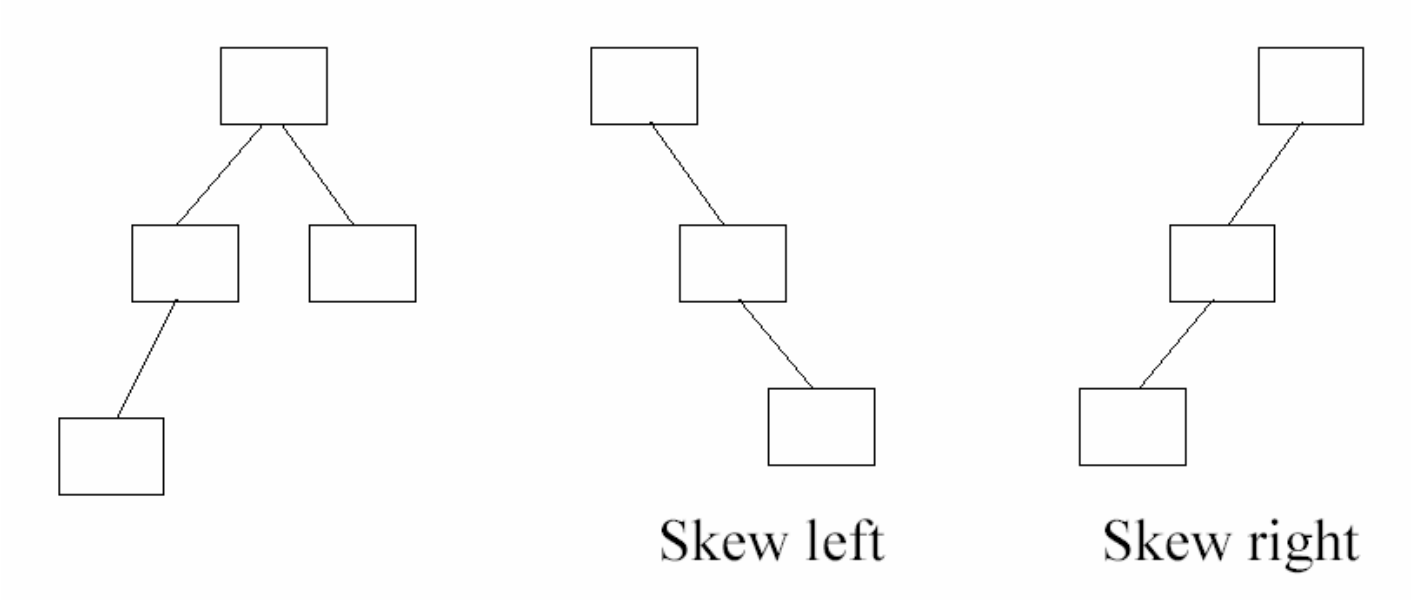
- A tree is a finite set of one or more nodes such that:
  - There is a specially designated node called the root
  - The remaining nodes are partitioned into n ≥ 0 disjoint sets T1, T2, … Tn where each set is a tree
- T1, T2, … Tn are called the subtree of the root
- N-airy tree, binary tree

Binary Tree

- A tree that any node can have at most two branches
- We distinguish between Left subtree and Right subtree
- A binary tree may have zero nodes (empty tree)
- A binary tree is different object than a tree
- Definition:
  - A binary tree is a finite set of nodes which is either empty or consists of a root and two disjoint binary tree called the left subtree and the right subtree

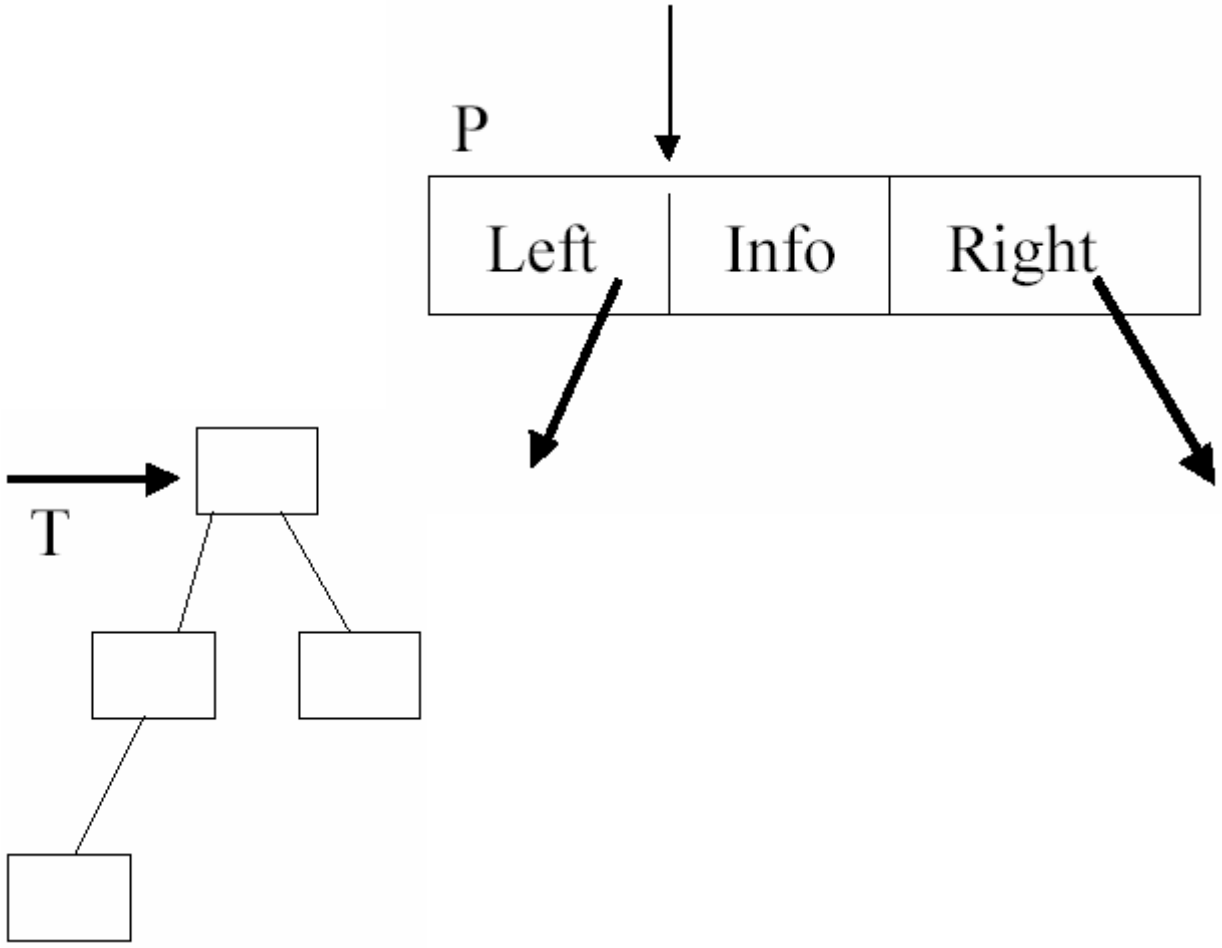
Binary Tree

- Definition:
  - A binary tree is a finite set of nodes which is either empty or consists of a root and two disjoint binary tree called the left subtree and the right subtree



A Binary Tree Node

- Left(P)
- Right(P)
- Info(P)
- Root(T)



Binary Tree Representation

- Contiguous representation by array
  - If P is a node, then Left(P)=P*2 and Right(P)=2*P+1
- Linked representation

Binary Tree

- Primitives:
  - Traversal: preorder, inorder, postorder
  - Searching
  - MakeTree
- Variations:
  - Search Tree
  - Balanced Tree
  - Threaded Tree

To Build A Tree

- Iterative methods
- Recursive methods

Data Structure Case Studies

- Polinom
- Occurrence of each letter in a text file
- Memory management
- Multilist
- Representation of N-M relation
- Topological sort