

# COMP0020 Functional Programming

## Lecture 15

<https://powcoder.com>  
Lists, Trees and Graphs

(link to implementation issues)

[Add WeChat powder](#)

# Contents

- Representation in memory : LISTS
- Trees
  - ▶ Representation in memory
  - ▶ Multiway branching trees
- Graphs
  - ▶ Source code
  - ▶ Representation in memory

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

## How lists are represented in memory

We cover basic mechanisms : in practice, functional language implementations (such as Miranda) may use different representations in different contexts (e.g. whether the list is a data value embedded in the program and known at compile-time, or whether it is constructed dynamically at run-time).

A very simple strategy is to manage physical memory locations in groups of three (called “cells”) :

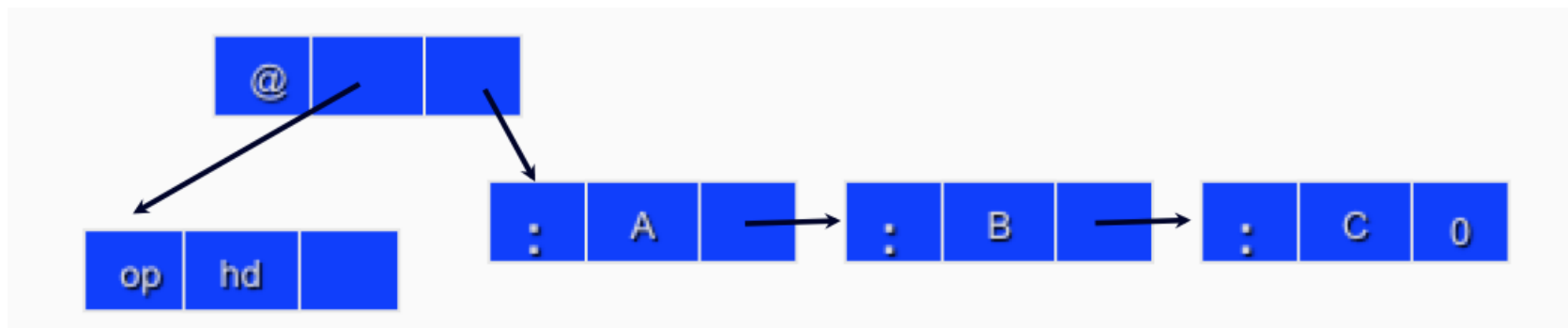
- A “tag” (indicating the kind of cell) followed by two other items (“fields”)
- Can be anywhere in memory

<https://powcoder.com>

```
x = ('A' : ('B' : ('C' : [])))
```

```
main = hd x
```

Add WeChat powcoder



# Representing TREES in memory

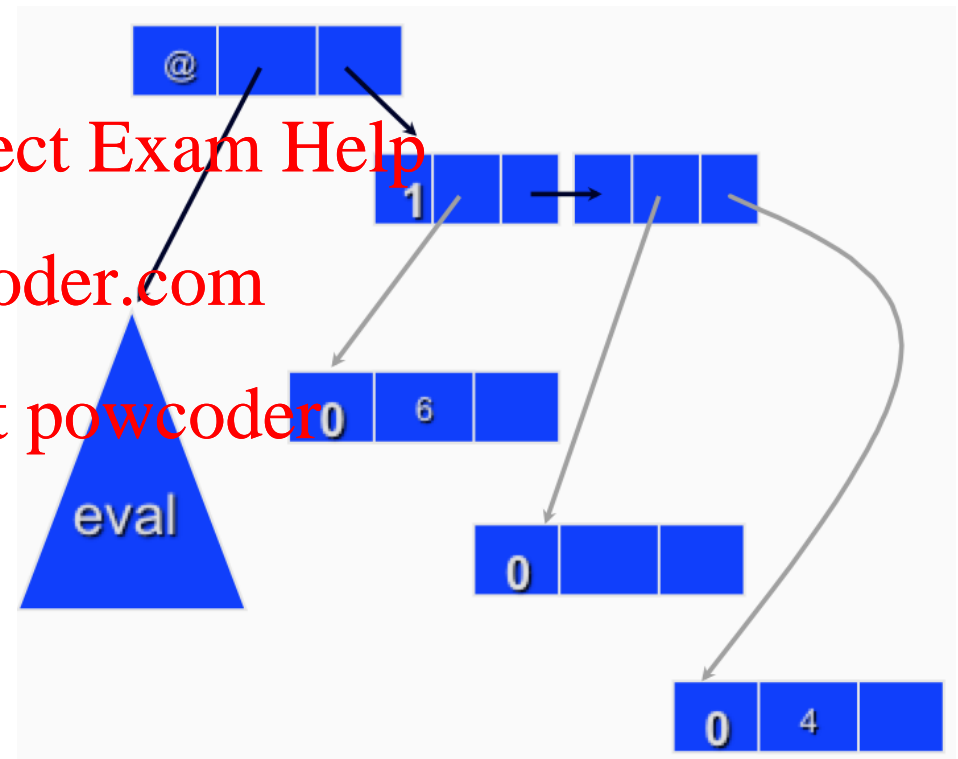
*exp ::=*  
     *Const num*  
     *| App exp op exp*  
     *| Bracketed exp*

*op ::=*   *Plus*   *| Minus* *| Times* *| Divide*

*x :: exp*

*x = App (Const 6) Plus (Const 4)*

*main = eval x*



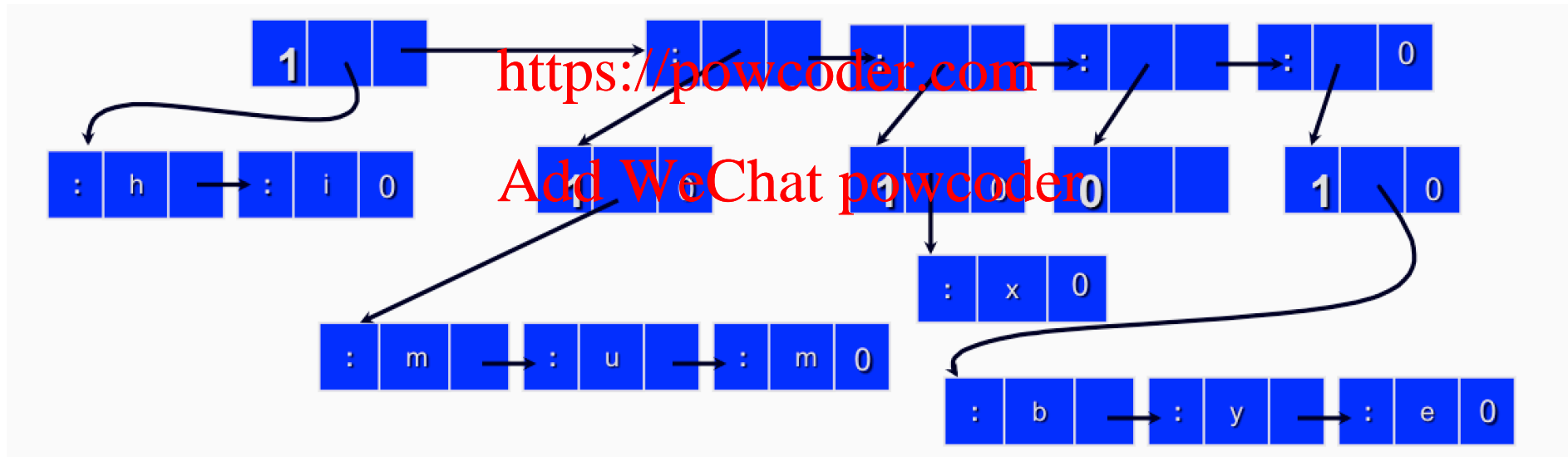
## Multiway branching trees

```
multitree * ::= Empty | Node * [multitree *]
```

```
x :: multitree [char]
```

```
x = Node "hi" [ (Node "mum" []), (Node "x" []), Empty, (Node "bye" []) ]
```

Assignment Project Exam Help



## Multiway branching trees

$\text{multitree } * : ::= \text{Empty} \mid \text{Node } * [\text{multitree } *]$

$x : : \text{multitree } [\text{char}]$

$x = \text{Node } \text{"hi"} [ (\text{Node } \text{"mum"} []), (\text{Node } \text{"x"} []), \text{Empty}, (\text{Node } \text{"bye"} []) ]$

## Assignment Project Exam Help

$\text{rightmost} :: \text{multitree } * \rightarrow *$

$\text{rightmost } \text{Empty} = \text{error "rightmost of empty tree"}$

$\text{rightmost } (\text{Node } x []) = x$

$\text{rightmost } (\text{Node } x \text{ ns}) = \text{rightmost } (\text{last } \text{ns1}), \text{ if } (\text{ns1} \sim= [])$

$= x, \text{ otherwise}$

where

$\text{ns1} = \text{filter } (\sim= \text{Empty}) \text{ ns}$

## Graphs : simple source code

```

multitree * ::= Empty      | Node * [multitree *]      || (as before)
a =      Node 'A'  [c, b]
b =      Node 'B'  [a, d]
c =      Node 'C'  [a, d]
d =      Node 'D'  []
graph :: multitree char
graph = a
firstlink  :: multitree *      — > multitree *
firstlink Empty                = error "firstlink of empty graph"
firstlink (Node x [])          = error "firstlink of node with no first link"
firstlink (Node x (n : ns))  = n

```

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

## Graphs : source code using a list

```

multitree * ::= Empty      | Node * [multitree *]      || (as before)
glist  ::  [multitree char]
glist = [ Node 'A' [ (glist ! 2), (glist ! 1) ],
          Node 'B'  [  (glist ! 3) ],
          Node 'C'  [ (glist ! 0), (glist ! 3) ],
          Node 'D'  [] ]
graph :: multitree char
graph = hd glist
firstlink  :: multitree *  — > multitree *
firstlink Empty           = error "firstlink of empty graph"
firstlink (Node x [])     = error "firstlink of node with no first link"
firstlink (Node x (n : ns)) = n

```



The diagram illustrates a linked list structure. It starts with a pointer labeled 'glist' pointing to the first node. The nodes are represented as boxes divided into three parts: a pointer, an integer, and a character. The first node's pointer points to the second node, and so on, forming a chain. The nodes contain the following data:

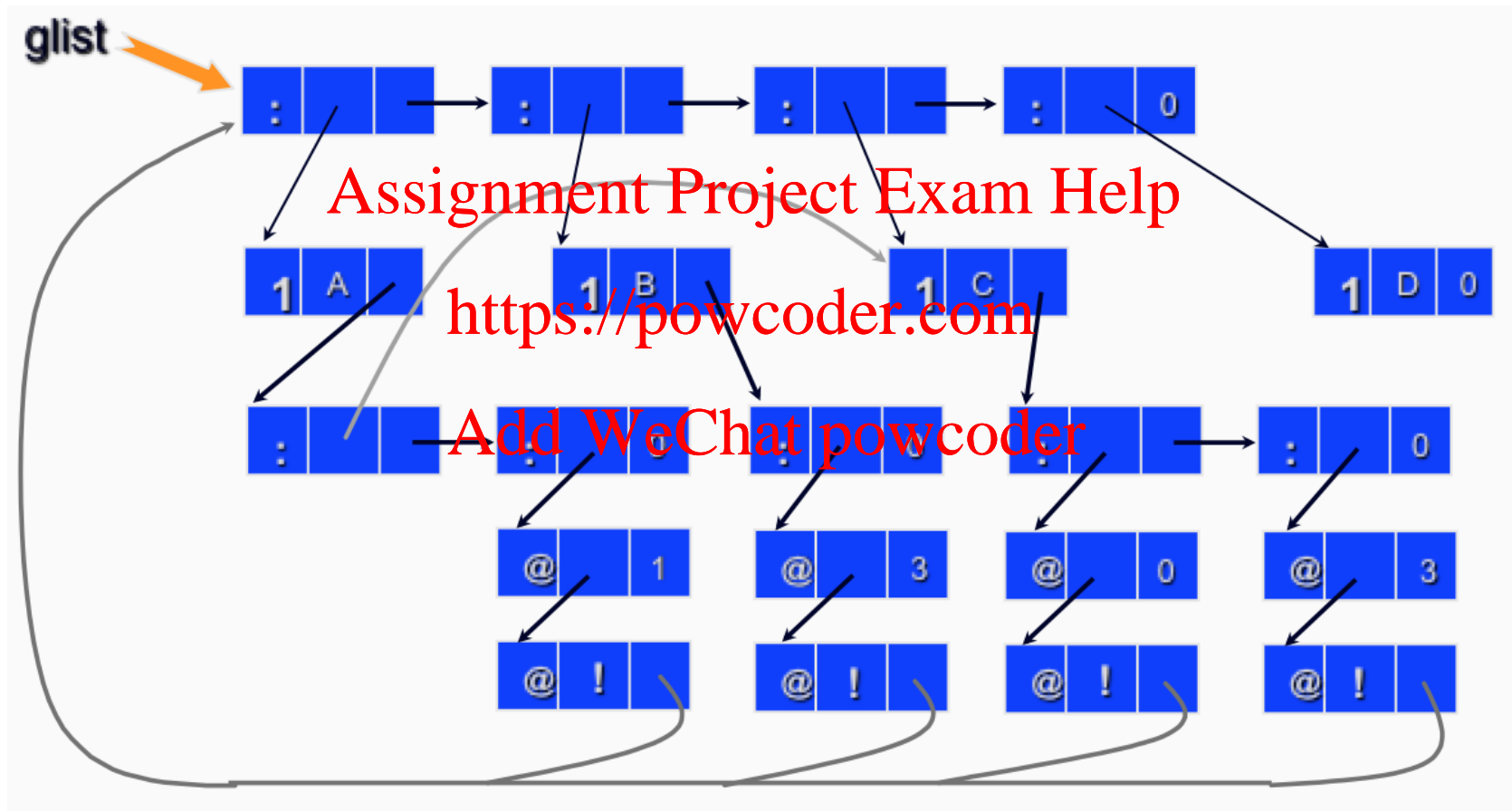
- Node 1: Pointer (empty), Integer (1), Character (A)
- Node 2: Pointer (empty), Integer (1), Character (B)
- Node 3: Pointer (empty), Integer (1), Character (C)
- Node 4: Pointer (empty), Integer (1), Character (D), followed by a null pointer (0).

Below the main chain, there are more nodes, some of which are highlighted in purple. These nodes contain the following data:

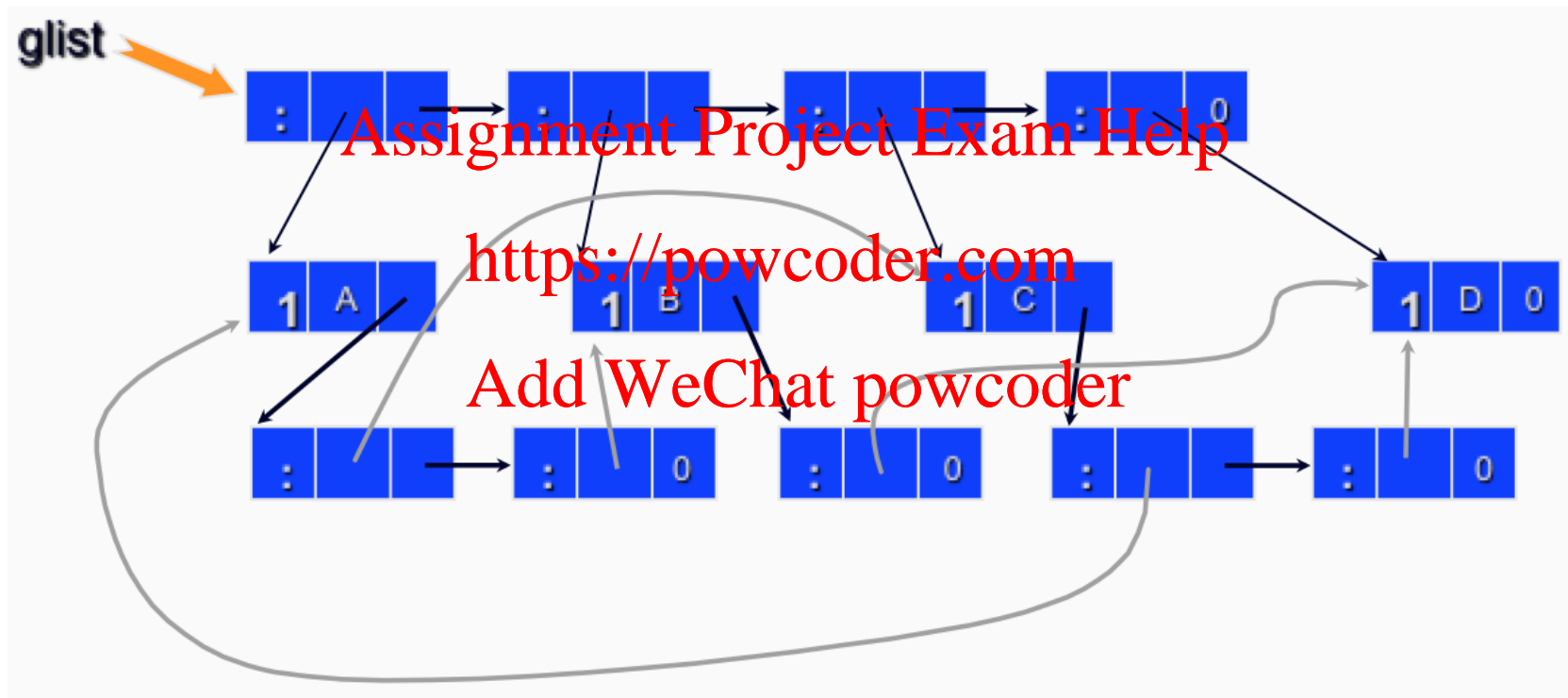
- Node 5: Pointer (empty), Integer (2), Character (@)
- Node 6: Pointer (empty), Integer (1), Character (@)
- Node 7: Pointer (empty), Integer (3), Character (@)
- Node 8: Pointer (empty), Integer (0), Character (@)
- Node 9: Pointer (empty), Integer (3), Character (@)

Arrows indicate the flow of pointers between nodes. A long arrow from the 'glist' pointer points to the first node. Other arrows show the sequence of nodes in the list. Some nodes have additional arrows pointing to them, possibly indicating a second list or a specific path.

## Graphs : representation after evaluating (firstlink graph)



## Graphs : representation after evaluating all links



# Summary

- Representation in memory : LISTS
- Trees
  - ▶ Representation in memory
  - ▶ Multiway branching trees
- Graphs
  - ▶ Source code
  - ▶ Representation in memory

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Assignment Project Exam Help

END OF LECTURE  
<https://powcoder.com>

Add WeChat powcoder