

COMP0020 Functional Programming

Lecture 15a

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

(link to implementation issues)
Add WeChat powcoder

Contents

Assignment Project Exam Help

<https://powcoder.com>

- Answer to student query :
- Writing a function to traverse a cyclic graph

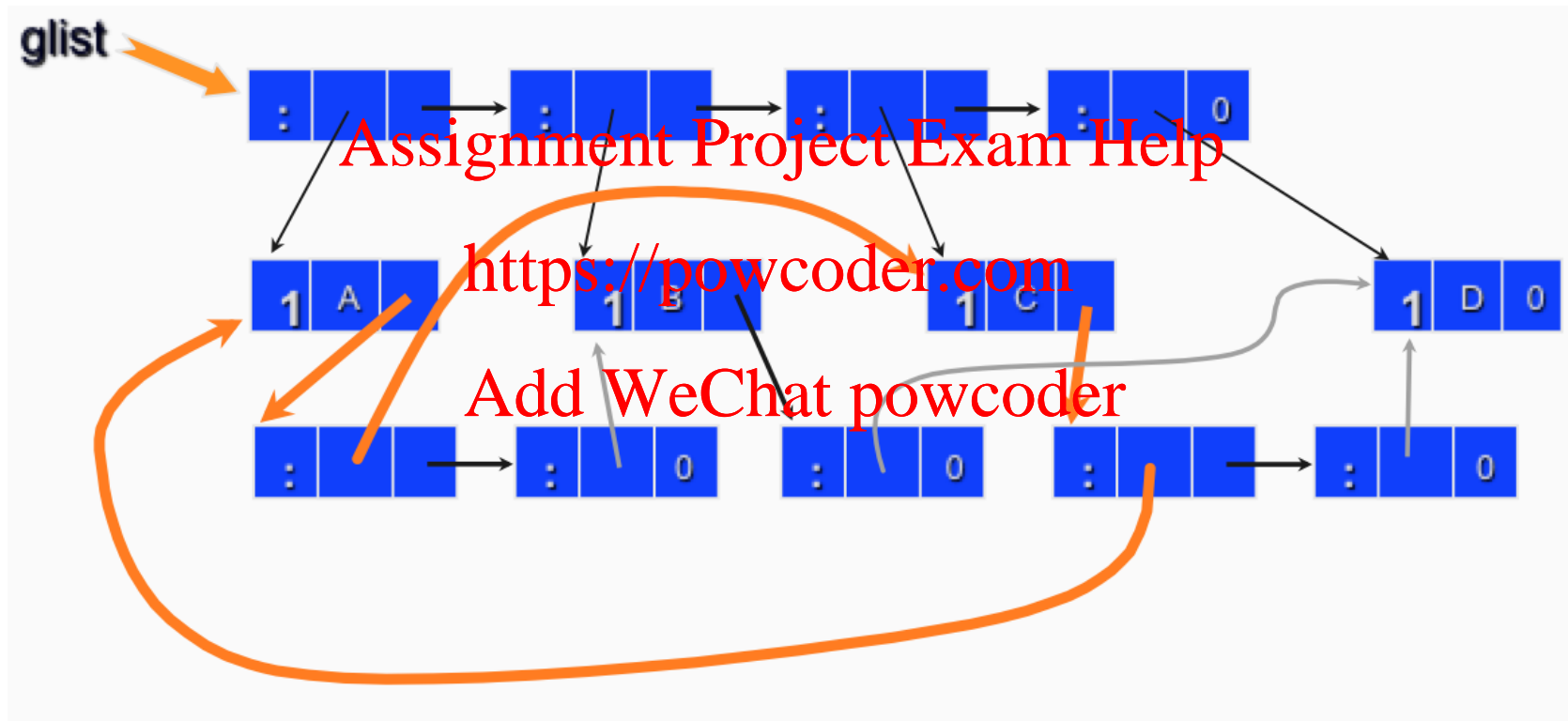
Add WeChat powcoder

Graphs : source code

Assignment Project Exam Help
<https://powcoder.com>
Add WeChat powcoder

```
multitree * ::= Empty | Node * [multitree *]  
graph :: multitree char  
graph = hd glist  
glist :: [multitree char]  
glist = [Node 'A' [(glist ! 2), (glist ! 1)], Node 'B' [(glist ! 3)],  
         Node 'C' [(glist ! 0), (glist ! 3)], Node 'D' []]
```

Graphs : representation after evaluating all links



Graphs : source code

```
printgraph :: multitree char -> [char]
```

```
printgraph Empty = []
```

```
printgraph (Node x gs) = "Node" ++ [x] ++ "[" ++ (xpg gs [x]) ++ "]"
```

Assignment Project Exam Help

```
xpg :: [multitree char] -> [char] -> [char]
```

```
xpg [] y = []
```

```
xpg [Empty] y = "Empty"
```

```
xpg (Empty : gs) y = "Empty," ++ (xpg gs y)
```

```
xpg ((Node x ns) : gs) y = "Node" ++ [x] ++ " seen" ++ s ++ (xpg gs y), if (member y x)
```

```
= "Node" ++ [x] ++ "[" ++ (xpg ns (x : y)) ++ z ++ (xpg gs (x : y)), otherwise
```

```
where s = "", if gs = []
```

```
= ", ", otherwise
```

```
z = "]" ++ s
```

<https://powcoder.com>

Add WeChat powcoder

Summary

Assignment Project Exam Help

<https://powcoder.com>

- Answer to student query : **Add WeChat powcoder**
- Writing a function to traverse a cyclic graph

Assignment Project Exam Help

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

Add WeChat powcoder

Assignment Project Exam Help

END OF ALL LECTURES
<https://powcoder.com>

Add WeChat powcoder