University of Asia Pacific

CSE 430

Compiler Design Lab

Assignment – 1

Write a Program for Symbol Table

Name: Md. Azim Islam

Registration: 19201026

Section: A2

Course Code: CSE 430

Semester: Spring 2023

Baivab Das

Lecturer, University of Asia Pacific, Dhaka

CODE

```
from functools import reduce
import sys

#Symbol Table
class Node:
    def __init__(self, name, type, size, dim, loc, addr):
        self.name = name
        self.type = type
        self.size = size
        self.dim = dim
        self.loc = loc
        self.addr = addr
        self.next = None

class LinkedList:
    def __init__(self) -> None:
        self.head = Node("0Head", 0, 0, 0 ,0 ,0)

    def add(self, node):
        t = self.getTail(node.name)
        if t:
            t.next = node
            node.next = None
        else:
            print("Error: A symbol already exists with the same
name!\nTry Updating the symbol.\n")

    def getTail(self, name):
        curr_node = self.head

        while curr_node.next:
            curr_node = curr_node.next
            if curr_node.name == name:
                return None

        return curr_node

    def search(self, name):
        curr_node = self.head
```

```
        while curr_node.next:
            curr_node = curr_node.next
            if curr_node.name == name:
                return curr_node
        else:
            return None
        return None

    def update(self, name, kwargs):
        node = self.search(name)
        if node:
            for k in kwargs:
                if k == 'name':
                    node.name = kwargs['name']
                if k == 'type':
                    node.type = kwargs['type']
                if k == 'size':
                    node.size = kwargs['size']
                if k == 'loc':
                    node.loc = kwargs['loc']
                if k == 'dim':
                    node.dim = kwargs['dim']
                if k == 'addr':
                    node.addr = kwargs['addr']
        else:
            print('Error NODE not found!')

    def delete(self, name):
        n1 = self.head
        n2 = n1.next
        while n2:
            if n2.name == name:
                n1.next = n2.next
            else:
                n1 = n2
                n2 = n2.next
    def print(self):
        node = self.head.next
        while node:
            print(node.name, node.type, node.size, node.dim, node.loc,
node.addr)
```

```
            print("Node Hash: ", Hash_Name(node.name), '\n')
            node = node.next

sym_table = [LinkedList() for i in range(1024)]


def Hash_Node(node):
    hash_value = reduce(lambda x, y : x*y, [ord(i) for i in
node.name])%1024
    return hash_value

def Hash_Node(node):
    hash_value = reduce(lambda x, y : x*y, [ord(i) for i in
node.name])%1024
    return hash_value

def Hash_Name(name):
    hash_value = reduce(lambda x, y : x*y, [ord(i) for i in name])%1024
    return hash_value

for line in sys.stdin:
    OP = line.split(', ')[0]
    #name, type, size, dim, loc, addr
    if OP == 'insert':
        name, type, size, dim, loc, addr = line.split(', ')[1: ]
        addr = addr.strip()
        node = Node(name, type, size, dim, loc, addr)
        hzh = Hash_Node(node)
        sym_table[hzh].add(node)

    if OP == 'show':
        print("-"*50)
        print("Full Symbol Table")
        print("-"*50)
        print("name, type, size, dim, loc, addr")
        print("-"*50)
        for i in range(1024):
            sym_table[i].print()
        print("-"*50)

    if OP == 'update':
        name, type, size, dim, loc, addr = line.split(', ')[1: ]
```

```
        addr = addr.strip()
        hzh = Hash_Name(name)
        d = {'name':name, 'type':type, 'size':size, 'dim':dim,
'loc':loc, 'addr':addr}
        sym_table[hzh].update(name, d)
```

INPUT

insert, x, ID, 2, 1, 5, 0x6dfed4
show,
insert, x, ID, 2, 1, 5, 0x6dfed4
show,
insert, y, ID4, 2, 3, 6, 0x6dfe23
show,
update, y, ID26, 19, 20, 1, 0x6622
show

OUTPUT

```
------------------------------------------------
                Full Symbol Table
------------------------------------------------
         name, type, size, dim, loc, addr
------------------------------------------------
               x ID 2 1 5 0x6dfed4
                  Node Hash: 120

------------------------------------------------
Error: A symbol already exists with the same name!
            Try Updating the symbol.

------------------------------------------------
                Full Symbol Table
------------------------------------------------
         name, type, size, dim, loc, addr
------------------------------------------------
               x ID 2 1 5 0x6dfed4
                  Node Hash: 120

------------------------------------------------
------------------------------------------------
                Full Symbol Table
------------------------------------------------
         name, type, size, dim, loc, addr
------------------------------------------------
               x ID 2 1 5 0x6dfed4
                  Node Hash: 120

              y ID4 2 3 6 0x6dfe23
                  Node Hash: 121

------------------------------------------------
------------------------------------------------
                Full Symbol Table
------------------------------------------------
         name, type, size, dim, loc, addr
------------------------------------------------
```

x ID 2 1 5 0x6dfed4

Node Hash: 120

y ID26 19 20 1 0x6622

Node Hash: 121
