University of Asia Pacific
Department of Computer Science & Engineering

Course Title: Compiler Design Lab
Course Code: CSE 430
Lab 2 Report

Submitted by:-
Junnatul Mawa
Section:-B1
Registration:- 20101070

Task: Construct a simple hash-based symbol table (data-dictionary) based on chaining.

Symbol table is an important data structure created and maintained by compilers in order to store
information about the occurrence of various entities such as variable names, function names,
objects, classes, interfaces etc. The information is collected by the analysis phase of the compiler
and used by dash synthesis phase to generate target code.

Usage of symbol table by all the phases of a compiler

i. Lexical analysis: Creates new entries for each new identifiers.

ii. Syntax Analysis: Adds information regarding attributes like type, scope, dimension, line
of reference and line of use.

iii. Semantic Analysis: adds information regarding attributes like type, scope, dimension, line
of reference and line of usage.

iv. Intermediate Code Generation: information in symbol table helps to add temporary
variable's information.

v. Code Optimization: information in symbol table used in machine-dependent optimization
by considering address and variable information.

vi. Target Code Generation: generates the code by using the address information of identifiers.

Symbol Table Entries

each entry in the symbol table is associated with "attributes" that support the compiler in
different phases the attributes are:

- Name
- Size
- Dimension (used if it is an array)
- Type
- Line of declaration (where the variable is declared to generate errors)

- Line of usage (link list to keep track of multiple usage of a variable)
- Address

Here is the code-

```
class SymbolTable:
    def __init__(self):
        self.table = {}

    def insert(self, name, type_, size, dimension, line_of_code, address):
        if name not in self.table:
            self.table[name] = (type_, size, dimension, line_of_code,
address)
            print("Inserted successfully.")
        else:
            print("Name already exists in the symbol table.")

    def search(self, name):
        if name in self.table:
            return self.table[name]
        else:
            return None

    def delete(self, name):
        if name in self.table:
            del self.table[name]
            print("Deleted successfully.")
        else:
            print("Name not found in the symbol table.")

    def update(self, name, type_, size, dimension, line_of_code, address):
        if name in self.table:
            self.table[name] = (type_, size, dimension, line_of_code,
address)
            print("Updated successfully.")
        else:
            print("Name not found in the symbol table.")
```

```
    def show_contents(self):
        print("Name\tType\tSize\tDimension\tLine of Code\tAddress")
        for name, values in self.table.items():
            print(f"{name}\t\t{values[0]}\t\t{values[1]}\t\t
{values[2]}\t\t\t{values[3]}\t\t\t{values[4]}")

    def get_hash_key(self, name, x):
        return hash(name) % x


def main():
    symbol_table = SymbolTable()
    x = int(input("Enter size of the hash table: "))

    while True:
        print("\nSymbol Table Operations:")
        print("1. Insert")
        print("2. Search")
        print("3. Delete")
        print("4. Update")
        print("5. Show Contents")
        print("6. Get Hash Key")
        print("7. Exit")

        choice = input("Enter your choice: ")

        if choice == '1':
            name = input("Enter name: ")
            type_ = input("Enter type: ")
            size = input("Enter size: ")
            dimension = input("Enter dimension: ")
            line_of_code = input("Enter line of code: ")
            address = input("Enter address: ")
            symbol_table.insert(name, type_, size, dimension,
line_of_code, address)

        elif choice == '2':
            name = input("Enter name to search: ")
            result = symbol_table.search(name)
            if result:
```

```
                print("Found:")
                print(result)
            else:
                print("Name not found in the symbol table.")

        elif choice == '3':
            name = input("Enter name to delete: ")
            symbol_table.delete(name)

        elif choice == '4':
            name = input("Enter name to update: ")
            type_ = input("Enter type: ")
            size = input("Enter size: ")
            dimension = input("Enter dimension: ")
            line_of_code = input("Enter line of code: ")
            address = input("Enter address: ")
            symbol_table.update(name, type_, size, dimension,
line_of_code, address)

        elif choice == '5':
            symbol_table.show_contents()

        elif choice == '6':
            name = input("Enter name to get hash key: ")
            hash_key = symbol_table.get_hash_key(name, x)
            print(f"Hash key for {name}: {hash_key}")

        elif choice == '7':
            print("Exiting program.")
            break

        else:
            print("Invalid choice. Please enter a valid option !")


if __name__ == "__main__":
    main()
```

Here is the output-

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

Enter type: variable
Enter size: 4
Enter dimension: 4
Enter line of code: 4
Enter address: 4
Inserted successfully.

Symbol Table Operations:
1. Insert
2. Search
3. Delete
4. Update
5. Show Contents
6. Get Hash Key
7. Exit
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                    Python

7. Exit
Enter your choice: 5
Name    Type    Size    Dimension       Line of Code    Address
mawa            variable                4               4                       4                       4

Symbol Table Operations:
1. Insert
2. Search
3. Delete
4. Update
5. Show Contents
6. Get Hash Key
7. Exit
Enter your choice: 5
Name    Type    Size    Dimension       Line of Code    Address
mawa            variable                4               4                       4                       4

Symbol Table Operations:
1. Insert
2. Search
3. Delete
4. Update
5. Show Contents
6. Get Hash Key
```