U18CO018

Shubham Shekhaliya

Assignment – 2

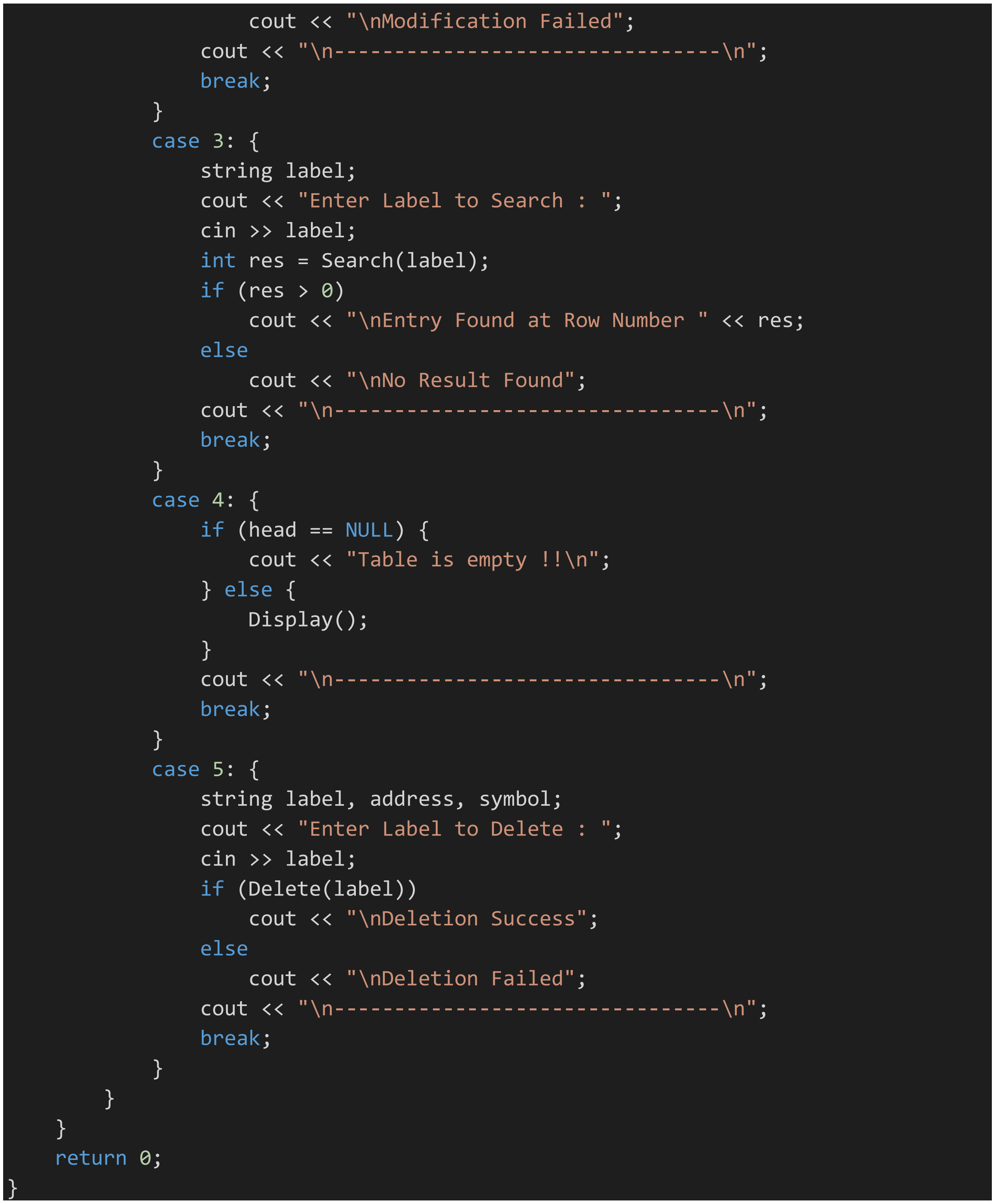
Subject – System Software

Write a dynamic program to generate a Symbol Table from the first pass assembler.

Code:-

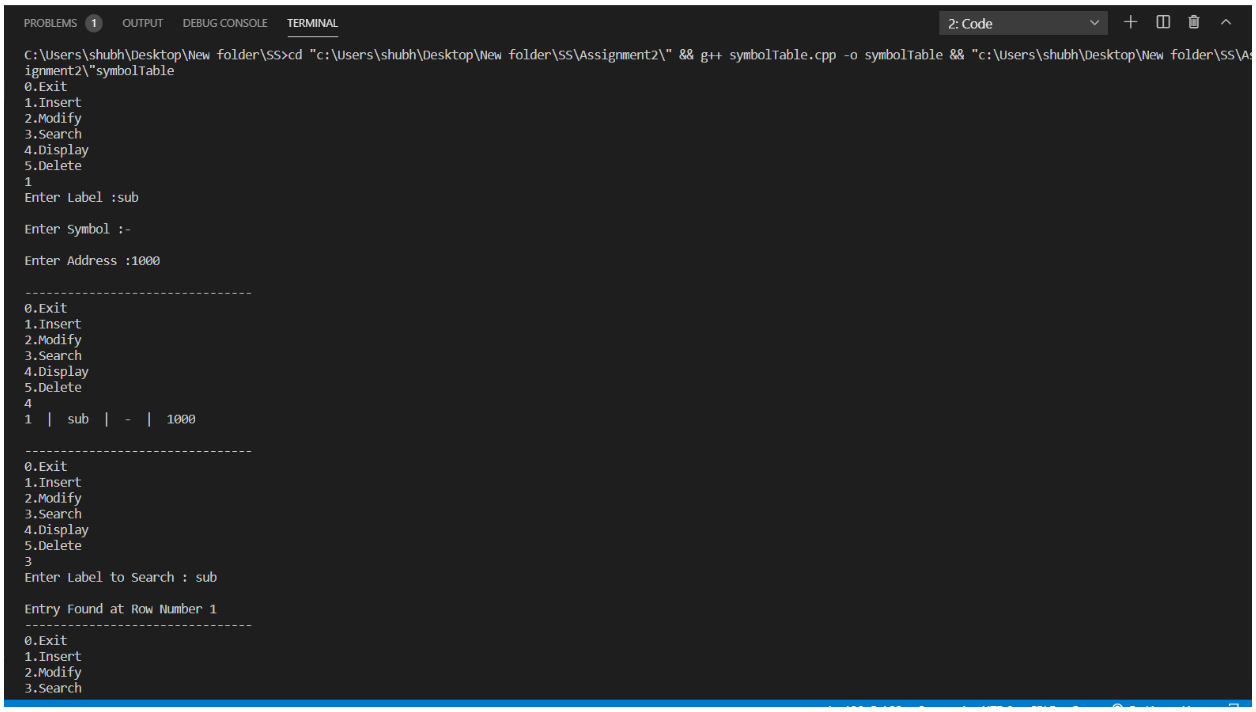
```
#include<iostream>
using namespace std;

struct Node {
    string label, symbol, address;
    struct Node* next;
    Node(string label,string symbol,string address) {
        this->label = label;
        this->symbol = symbol;
        this->address = address;
        this->next = NULL;
    }
};

Node* head=NULL,*tail=NULL;

//Insert function to add Row in the Table
void Insert(string label,string symbol,string address) {
    Node *now = new Node(label, symbol, address);
    if (head == NULL) {
        head = now;
        tail = now;
        return;
    }
    tail->next = now;
    tail = tail->next;
}

//modify function to modify Row symbol based on label in the Table
bool Modify(string label,string symbol,string address) {
    Node *cur = head;
    while (cur) {
```

```cpp
        if (cur->label == label) {
            cur->symbol = symbol;
            cur->address = address;
            return true;
        }
        cur = cur->next;
    }
    return false;
}

//Search function to search Row based on label in the Table
int Search(string label) {
    Node *cur = head;
    int cnt = 1;
    while (cur) {
        if (cur->label == label)
            return cnt;
        cur = cur->next;
        cnt++;
    }
    return -1;
}

void Display() {
    Node *cur = head;
    int cnt = 1;
    while (cur) {
        cout << cnt << "  |  " << cur->label << "  |  " << cur-
>symbol << "  |  " << cur->address << "\n";
        cur = cur->next;
        cnt++;
    }
}
//Delete function to delete Row in the Table
bool Delete(string label) {
    if (head->label == label) {
        Node *tp = head;
        head = head->next;
        free(tp);
        return true;
    }
    Node *cur = head;
    while (cur->next) {
        if (cur->next->label == label) {
            Node *tp = cur->next;
            cur->next = cur->next->next;
            free(tp);
            return true;
        }
        cur = cur->next;
    }
    return false;
}

int main() {
    int op;
    while (true) {
        cout << "0.Exit\n";
        cout << "1.Insert\n";
        cout << "2.Modify\n";
        cout << "3.Search\n";
        cout << "4.Display\n";
        cout << "5.Delete\n";
        cin >> op;
        if (!op)
            break;
        switch (op) {
            case 1: {
                string label, address, symbol;
                cout << "Enter Label :";
                cin >> label;
                cout << "\nEnter Symbol :";
                cin >> symbol;
                cout << "\nEnter Address :";
                cin >> address;
                Insert(label, symbol, address);
                cout << "\n--------------------------------\n";
                break;
            }
            case 2: {
                string label, address, symbol;
                cout << "Enter Label to Modify : ";
                cin >> label;
                cout << "\nEnter New Symbol : ";
                cin >> symbol;
                cout << "\nEnter New Address : ";
                cin >> address;
                if (Modify(label, symbol, address))
                    cout << "\nModification Success";
                else
```

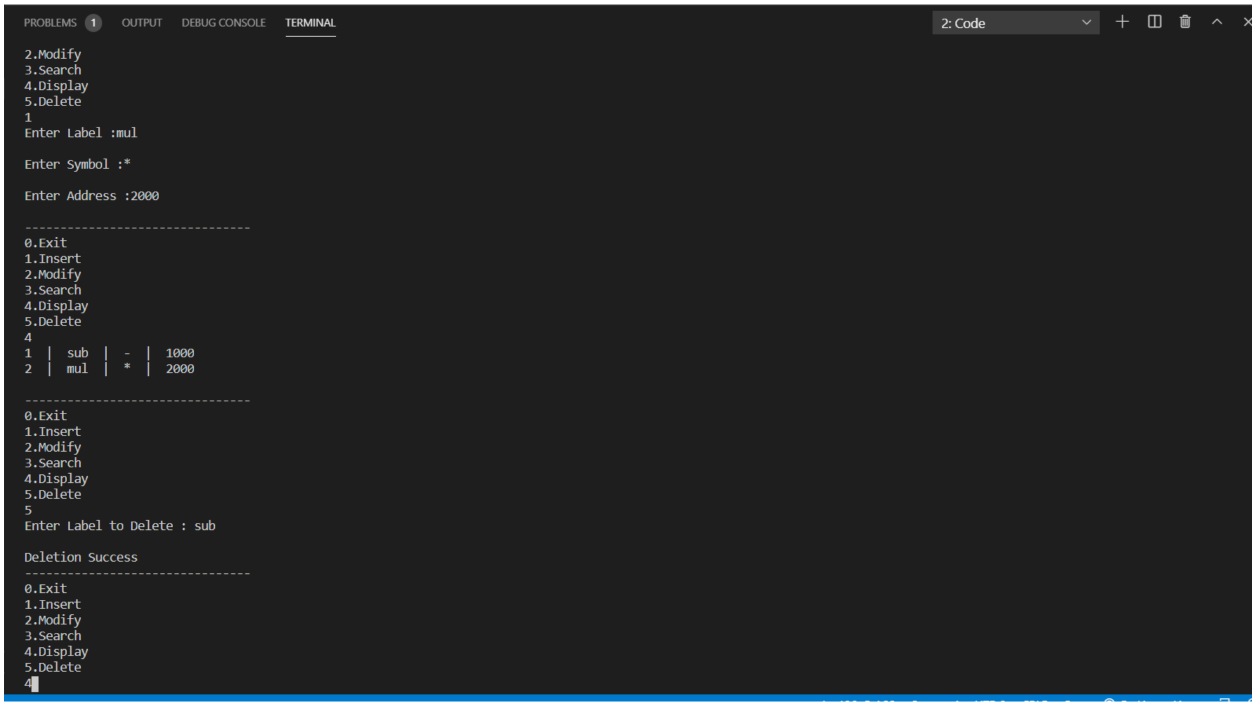


```
                    cout << "\nModification Failed";
                cout << "\n--------------------------------\n";
                break;
            }
            case 3: {
                string label;
                cout << "Enter Label to Search : ";
                cin >> label;
                int res = Search(label);
                if (res > 0)
                    cout << "\nEntry Found at Row Number " << res;
                else
                    cout << "\nNo Result Found";
                cout << "\n--------------------------------\n";
                break;
            }
            case 4: {
                if (head == NULL) {
                    cout << "Table is empty !!\n";
                } else {
                    Display();
                }
                cout << "\n--------------------------------\n";
                break;
            }
            case 5: {
                string label, address, symbol;
                cout << "Enter Label to Delete : ";
                cin >> label;
                if (Delete(label))
                    cout << "\nDeletion Success";
                else
                    cout << "\nDeletion Failed";
                cout << "\n--------------------------------\n";
                break;
            }
        }
    }
    return 0;
}
```

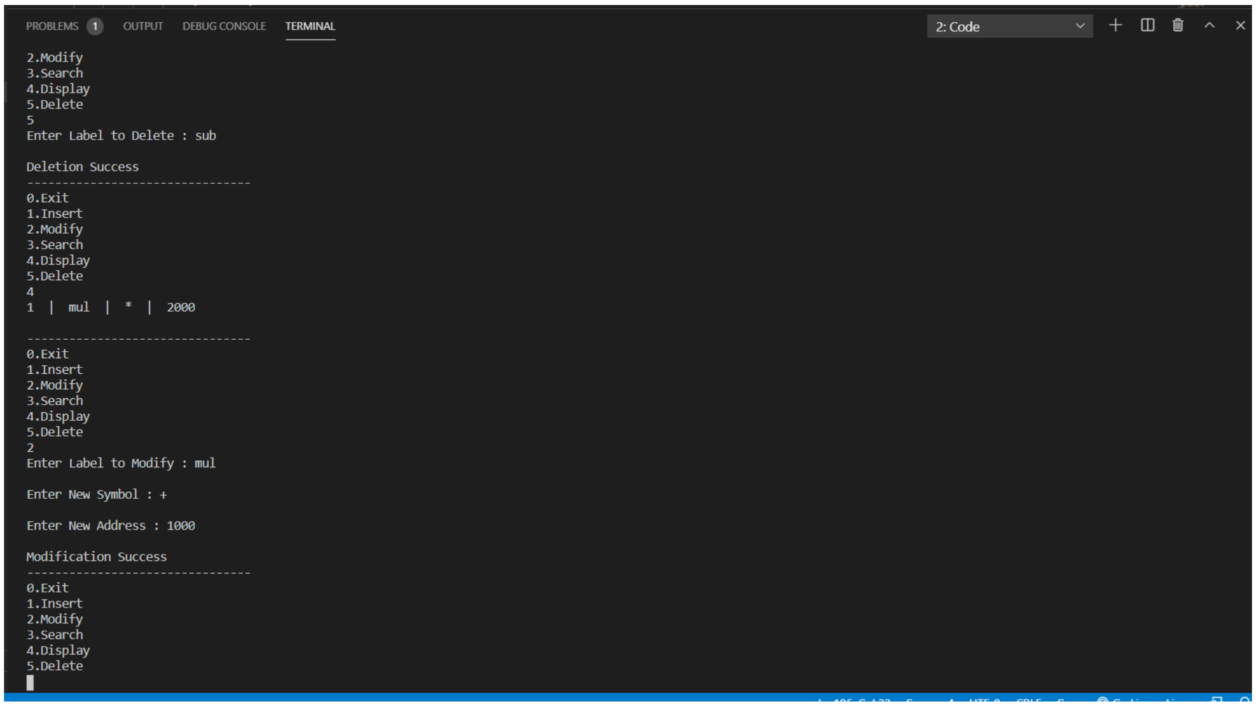
Output:-



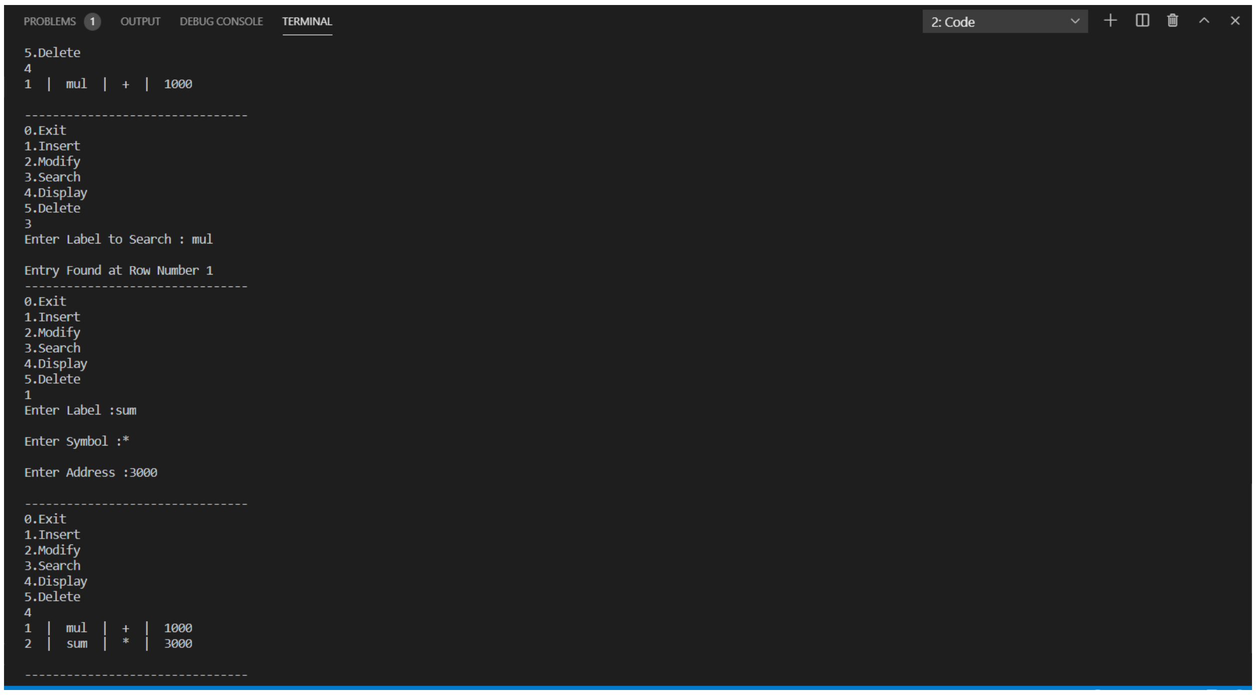
```
C:\Users\shubh\Desktop\New folder\SS>cd "c:\Users\shubh\Desktop\New folder\SS\Assignment2\" && g++ symbolTable.cpp -o symbolTable && "c:\Users\shubh\Desktop\New folder\SS\As
ignment2\"symbolTable
0.Exit
1.Insert
2.Modify
3.Search
4.Display
5.Delete
1
Enter Label :sub

Enter Symbol :-

Enter Address :1000

--------------------------------
0.Exit
1.Insert
2.Modify
3.Search
4.Display
5.Delete
4
1  |  sub  |  -  |  1000

--------------------------------
0.Exit
1.Insert
2.Modify
3.Search
4.Display
5.Delete
3
Enter Label to Search : sub

Entry Found at Row Number 1
--------------------------------
0.Exit
1.Insert
2.Modify
3.Search
```



```
2.Modify
3.Search
4.Display
5.Delete
1
Enter Label :mul

Enter Symbol :*

Enter Address :2000

--------------------------------
0.Exit
1.Insert
2.Modify
3.Search
4.Display
5.Delete
4
1  |  sub  |  -  |  1000
2  |  mul  |  *  |  2000

--------------------------------
0.Exit
1.Insert
2.Modify
3.Search
4.Display
5.Delete
5
Enter Label to Delete : sub

Deletion Success
--------------------------------
0.Exit
1.Insert
2.Modify
3.Search
4.Display
5.Delete
4
```



```
2.Modify
3.Search
4.Display
5.Delete
5
Enter Label to Delete : sub

Deletion Success
--------------------------------
0.Exit
1.Insert
2.Modify
3.Search
4.Display
5.Delete
4
1  |  mul  |  *  |  2000

--------------------------------
0.Exit
1.Insert
2.Modify
3.Search
4.Display
5.Delete
2
Enter Label to Modify : mul

Enter New Symbol : +

Enter New Address : 1000

Modification Success
--------------------------------
0.Exit
1.Insert
2.Modify
3.Search
4.Display
5.Delete
```



```
5.Delete
4
1  |  mul  |  +  |  1000

--------------------------------
0.Exit
1.Insert
2.Modify
3.Search
4.Display
5.Delete
3
Enter Label to Search : mul

Entry Found at Row Number 1
--------------------------------
0.Exit
1.Insert
2.Modify
3.Search
4.Display
5.Delete
1
Enter Label :sum

Enter Symbol :*

Enter Address :3000

--------------------------------
0.Exit
1.Insert
2.Modify
3.Search
4.Display
5.Delete
4
1  |  mul  |  +  |  1000
2  |  sum  |  *  |  3000

--------------------------------
```