ASSIGNMENT 8
18/04/23

NAME : SHRESTH SONKAR
REGNO : 20214272
GROUP : CS4D
TOPIC : AUTOMATA LAB
CODE : CS-14203

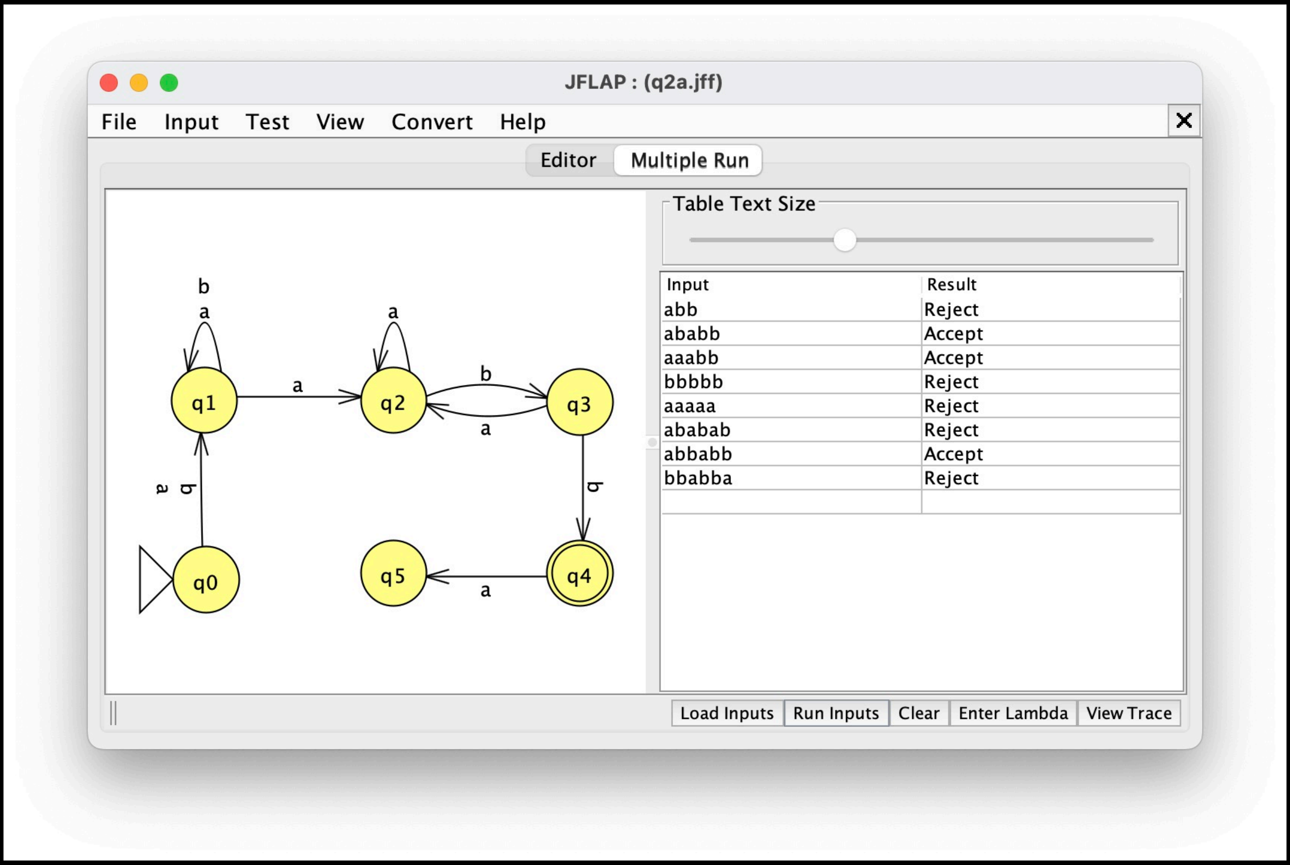
Q1

```
#include<stdio.h>

main() {
    int s, i, j, k, flag1 = 0, flag2 = 0;
    printf("Enter the no. of states: ");
    scanf("%d", &s);
    int mat[s][5], count = 0;
    int ar[s][2];
    int asso[s + 1];

    printf("Enter the matrix of states & outputs for
different inputs: \n");
    for (i = 0; i < s; i++) {
        for (j = 0; j < 5; j++) {
            scanf("%d", &mat[i][j]);
        }
    }

    for (k = 1; k <= s; k++) {
        for (i = 1; i < 5; i += 2) {
            for (j = 0; j < 4; j++) {
                if (mat[j][i] == k && mat[j][i + 1] ==
0) {
                    flag1 = 1;
                    asso[k] = 0;
                }
                if (mat[j][i] == k && mat[j][i + 1] ==
1) {
                    flag2 = 1;
                    asso[k] = 1;
                }
            }

        }
        if (flag1 == 1 && flag2 == 1) {
            ar[k - 1][0] = k;
            ar[k - 1][1] = 2;
        } else {
            ar[k - 1][0] = k;
            ar[k - 1][1] = 1;
        }
        flag1 = 0;
        flag2 = 0;
    }

    int sum = 0;
    for (i = 0; i < s; i++) {
        printf("%d %d\n", i + 1, ar[i][1]);
        sum += ar[i][1];
    }
    printf("\n Total new states will be %d", sum);

    int mat2[sum][5], l;
    k = 0;

    for (i = 0; i <
                sum; i++) {
        count = ar[mat[k][0] - 1][1];
        while (count) {
            for (j = 0, l = 0; j < 5; j++, l++) {
                mat2[i][j] = mat[k][l];
            }
            count--;
            if (count == 0) {
                k++;
            } else i++;
        }
    }


    for (i = 0; i <
                sum; i++) {
        for (j = 1; j < 5; j += 2) {
            if (ar[mat2[i][j] - 1][1] != 1) {
                mat2[i][j] = (mat2[i][j] * 10) +
mat2[i][j + 1];
            }
        }
    }

    for (i = 0; i <
                sum; i++) {
        if (ar[mat2[i][0] - 1][1] != 1) {
            mat2[i][0] = mat2[i][0] * 10;
            i++;
            mat2[i][0] = (mat2[i][0] * 10) + 1;;
        }
    }

    printf("\n The new table is: \n");
    for (i = 0; i < sum; i++) {
        printf("\n");
        for (j = 0; j < 5; j++)
            printf("%d ", mat2[i][j]);
    }

    int momat[sum][4];
    for (i = 0; i < 2; i++) {
        for (j = 0; j < sum; j++) {
            momat[j][i] = mat2[j][i];
        }
    }
    for (i = 3, j = 0; j < sum; j++)
        momat[j][i - 1] = mat2[j][i];

    for (i = 3, j = 0; j < sum; j++) {
        if (momat[j][0] >= 10)
            momat[j][i] = momat[j][0] % 10;
        else if (momat[j][0] < 10) {
            momat[j][i] = asso[momat[j][0]];
        }
    }

    printf("\n The moore table is: \n");

    if (momat[0][3] == 1) {
        printf("s ");
        for (j = 1; j < 3; j++)
            printf("%d ", momat[0][j]);
        printf("%d", 0);
    }

    for (i = 0; i < sum; i++) {
        printf("\n");
        for (j = 0; j < 4; j++)
            printf("%d ", momat[i][j]);
    }
}
```

Q2

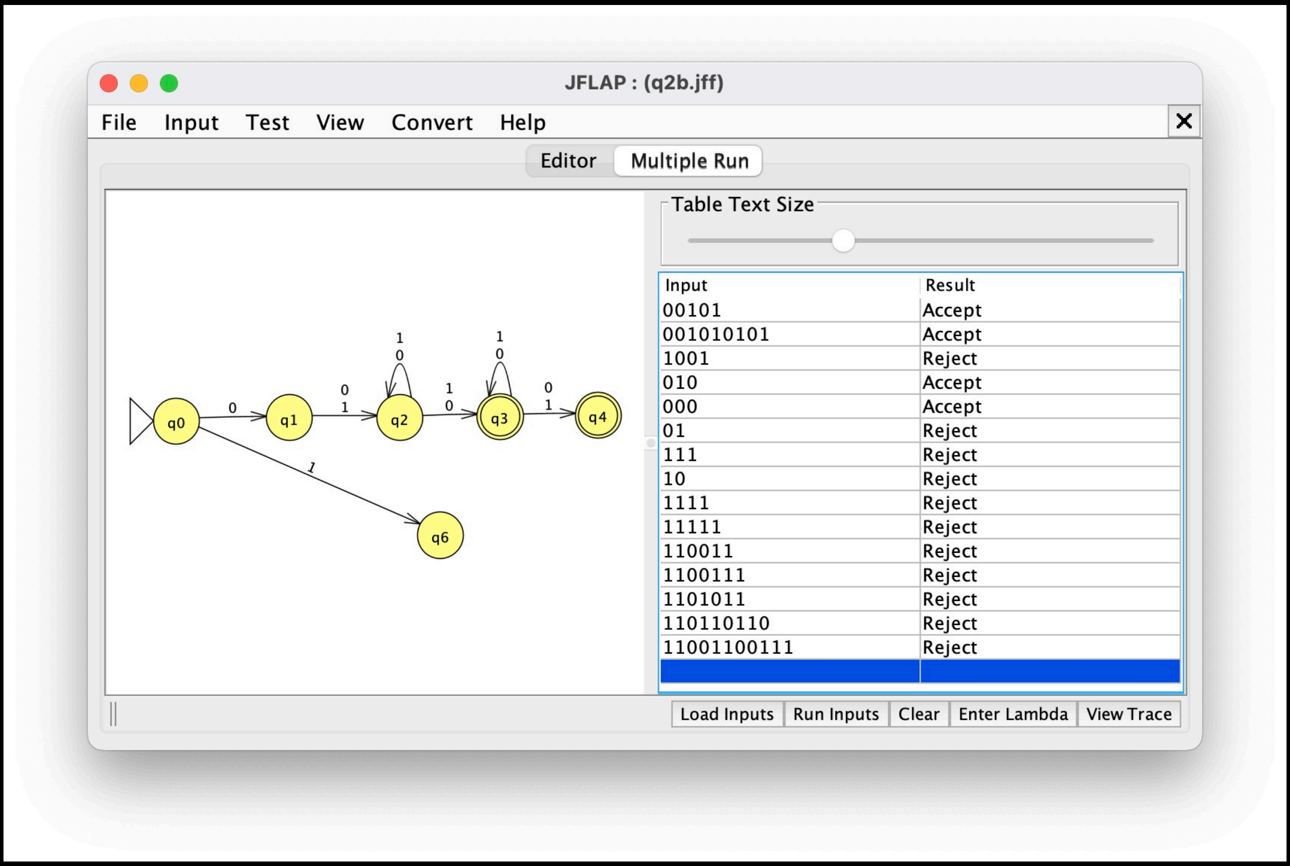
```
#include<stdio.h>
#include<string.h>

int i, j, k, l, m, n = 0, o, p, nv, z = 0, t, x = 0;
char str[10], temp[10], temp2[10], temp3[10];

struct prod {
    char lhs[10], rhs[10][10];
    int n;
} pro[10];

void findter() {
    for (k = 0; k < n; k++) {
        if (temp[i] == pro[k].lhs[0]) {
            for (t = 0; t < pro[k].n; t++) {
                for (x = 0; x < 10; x++)
                    temp2[x] = '\0';
                for (l = i + 1; l < strlen(temp); l++)
                    temp2[l - i - 1] = temp[l];
                temp[i] = '\0';
                for (l = 0; l < strlen(pro[k].rhs[t]);
l++)
                    temp[i + l] = pro[k].rhs[t][l];
                strcat(temp, temp2);
                if (str[i] == temp[i])
                    return;
            }
        }
    }

}

void main() {
    FILE *f;

    for (i = 0; i < 10; i++)
        pro[i].n = 0;

    f = fopen("tab3.txt", "r");
    while (!feof(f)) {
        fscanf(f, "%s", pro[n].lhs);
        if (n > 0) {
            if (strcmp(pro[n].lhs, pro[n - 1].lhs) ==
0) {
                pro[n].lhs[0] = '\0';
                fscanf(f, "%s", pro[n - 1].rhs[pro[n -
1].n]);
                pro[n - 1].n++;
                continue;
            }
        }
        fscanf(f, "%s", pro[n].rhs[pro[n].n]);
        pro[n].n++;
        n++;
    }
    n--;

    printf("\n\nTHE GRAMMAR IS AS FOLLOWS\n\n");
    for (i = 0; i < n; i++)
        for (j = 0; j < pro[i].n; j++)
            printf("%s -> %s\n", pro[i].lhs,
pro[i].rhs[j]);

    o = 0;
    for (i = 0; i < n; i++) {
        for (j = 0; j < pro[i].n; j++)
            if (pro[i].rhs[j][0] >= 65 && pro[i].rhs[j]
[0] <= 90) {
                o = 1;
                break;
            }
        if (o == 1)
            break;
    }
    if (i == n)
        printf("\n\nTHE GRAMMAR is a REGULAR
GRAMMAR !!!");
    else {
        printf("\n\nTHE GRAMMAR is NOT a REGULAR
GRAMMAR !!!");
        exit(1);
    }

    while (1) {
        for (x = 0; x < 10; x++)
            str[x] = '\0';
        printf("\n\nENTER ANY STRING ( 0 for EXIT ) :
");
        scanf("%s", str);
        if (str[0] == '0')
            exit(1);

        for (j = 0; j < pro[0].n; j++) {
            for (x = 0; x < 10; x++)
                temp[x] = '\0';
            strcpy(temp, pro[0].rhs[j]);

            m = 0;
            for (i = 0; i < strlen(str); i++) {
                if (str[i] == temp[i])
                    m++;
                else if (str[i] != temp[i] && temp[i]
>= 65 && temp[i] <= 90) {
                    findter();
                    if (str[i] == temp[i])
                        m++;
                }
            }
            for (x = 0; x < 10; x++)
                temp3[x] = '\0';
            strcpy(temp3, temp);
            temp3[strlen(temp) - 1] = '\0';
            if (m == strlen(str) && strcmp(temp3, str)
== 0 && strlen(temp3) != 1) {
                printf("\n\nTHE STRING can be
PARSED !!!");
                break;
            }
            if (m == strlen(str) && strlen(str) ==
strlen(temp)) {
                printf("\n\nTHE STRING can be
PARSED !!!");
                break;
            }
        }

        if (j == pro[0].n)
            printf("\n\nTHE STRING can NOT be
PARSED !!!");
    }
    printf("\n\n");
}
```

Q3a



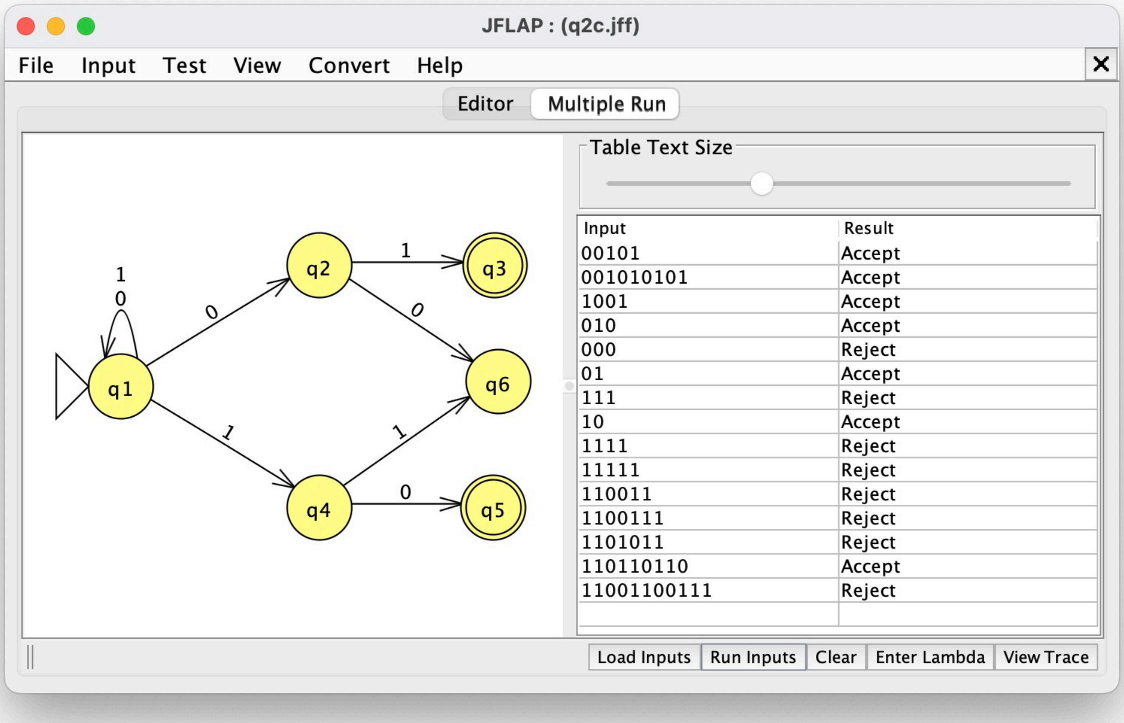
Input	Result
abb	Reject
ababb	Accept
aaabb	Accept
bbbbb	Reject
aaaaa	Reject
ababab	Reject
abbabb	Accept
bbabba	Reject

Q3b



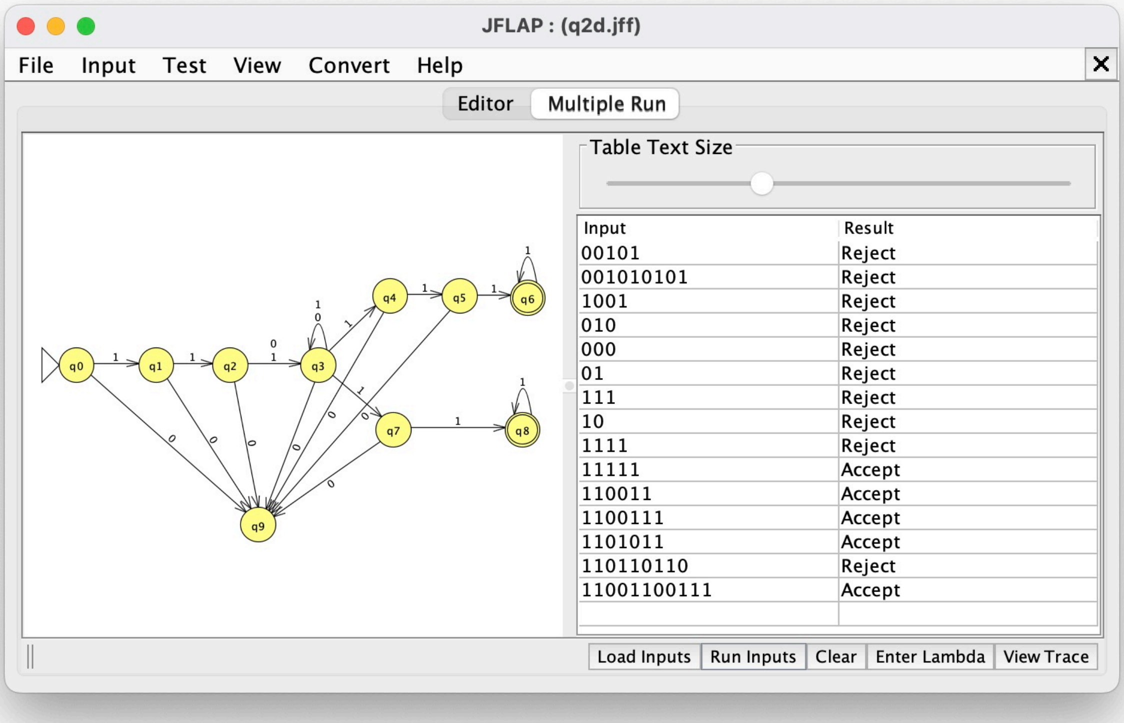
Input	Result
00101	Accept
001010101	Accept
1001	Reject
010	Accept
000	Accept
01	Reject
111	Reject
10	Reject
1111	Reject
11111	Reject
110011	Reject
1100111	Reject
1101011	Reject
110110110	Reject
11001100111	Reject

Q3c



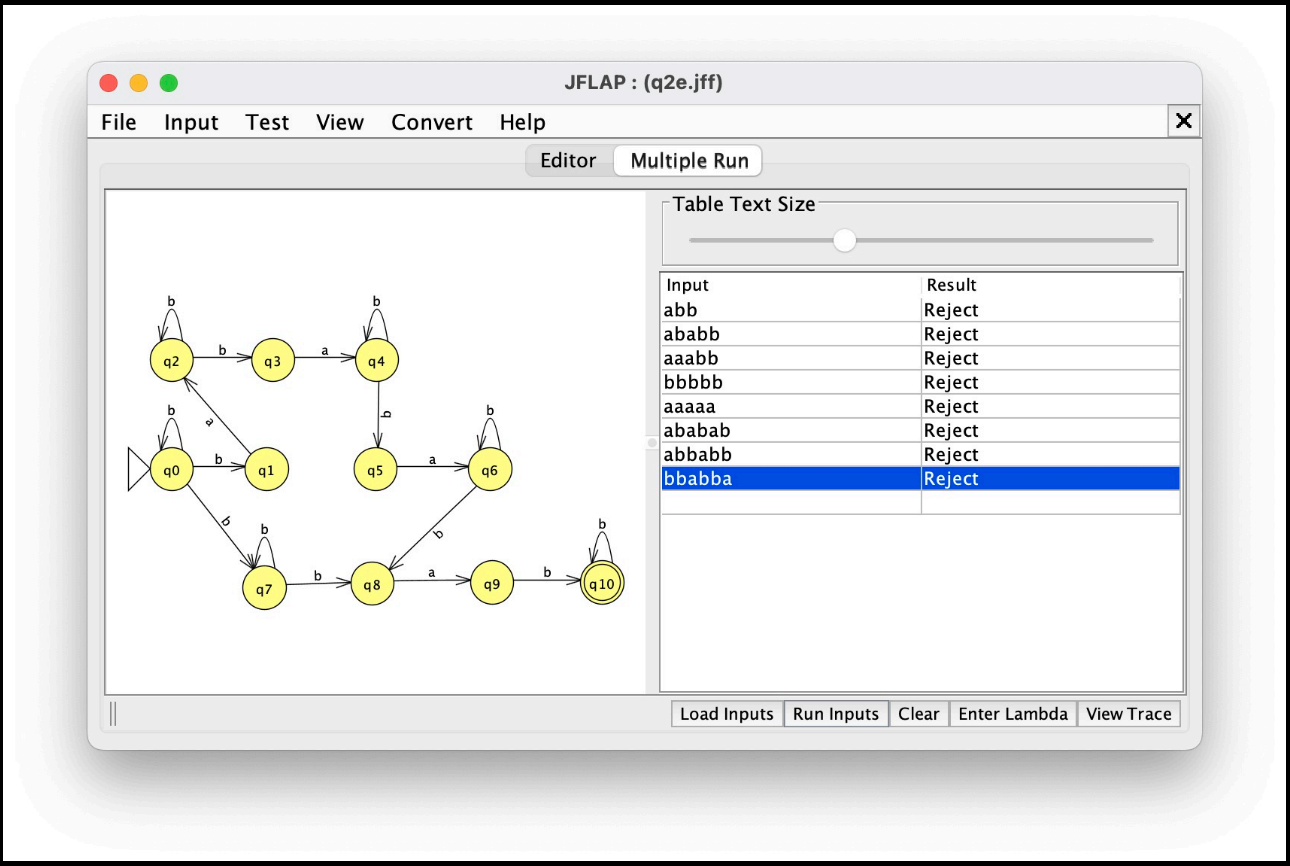
Input	Result
00101	Accept
001010101	Accept
1001	Accept
010	Accept
000	Reject
01	Accept
111	Reject
10	Accept
1111	Reject
11111	Reject
110011	Reject
1100111	Reject
1101011	Reject
110110110	Accept
11001100111	Reject

Q3d



Input	Result
00101	Reject
001010101	Reject
1001	Reject
010	Reject
000	Reject
01	Reject
111	Reject
10	Reject
1111	Reject
11111	Accept
110011	Accept
1100111	Accept
1101011	Accept
110110110	Reject
11001100111	Accept

Q3e



Input	Result
abb	Reject
ababb	Reject
aaabb	Reject
bbbbb	Reject
aaaaa	Reject
ababab	Reject
abbabb	Reject
bbabba	Reject