DS Assignment 8

Name: Himani Verma

Admission No: U19CS075

Ques: Implement Vector Clock

Here for

Event[i][j]=k

i=process id

j= event number

+k = I is sending vector to k

-k= I is receiving vector from |k|

Source Code:

```c
#include <stdio.h>

int max(int a, int b){
    if (a > b)
        return a;
    return b;
}
int abs(int a){
    if (a >= 0)
        return a;
    return a * (-1);
}

int main(){

    int event[3][10];
    int len[3];
    int i;
    for (int i = 1; i <= 3; i++)
    {
        printf("Enter number of events in process %d: ", i);
        scanf("%d", &len[i - 1]);

        printf("Enter sequence of events: ");
        int j;
        for (j = 0; j < len[i - 1]; j++)
        {
            scanf("%d", &event[i - 1][j]);
        }
    }
    i = 1;
    int j = 1, k = 1;
    int vector[3][10][3] = {0};
```

```
    while (i <= len[0] && j <= len[1] && k <= len[2]){
        //printf("%d %d %d\n", i, j, k);
        while (i <= len[0] && event[0][i] == 0){
            vector[0][i][0] = vector[0][i - 1][0] + 1;
            vector[0][i][1] = vector[0][i - 1][1];
            vector[0][i][2] = vector[0][i - 1][2];
            i++;
        }
        while (j <= len[1] && event[1][j] == 0){
            vector[1][j][0] = vector[1][j - 1][0];
            vector[1][j][1] = vector[1][j - 1][1] + 1;
            vector[1][j][2] = vector[1][j - 1][2];
            j++;
        }
        while (k <= len[2] && event[2][k] == 0){
            vector[2][k][0] = vector[2][k - 1][0];
            vector[2][k][1] = vector[2][k - 1][1];
            vector[2][k][2] = vector[2][k - 1][2] + 1;
            k++;
        }
        if (i <= len[0] && j <= len[1] && abs(event[0][i]) == 2 && abs(event[1][j]) == 1){

            vector[0][i][0] = vector[0][i - 1][0] + 1;

            vector[1][j][1] = vector[1][j - 1][1] + 1;
            vector[0][i][1] = vector[1][j][1];
            vector[1][j][0] = vector[0][i][0];
            if (event[0][i] < event[1][j]){
                vector[1][j][2] = vector[1][j - 1][2];
                vector[1][j][0] = vector[1][j - 1][0];
                vector[0][i][2] = max(vector[0][i - 1][2], vector[1][j][2]);
            }
            else{

                vector[0][i][2] = vector[0][i - 1][2];
                vector[0][i][1] = vector[0][i - 1][1];
                vector[1][j][2] = max(vector[1][j - 1][2], vector[0][i][2]);
            }
            i++;
            j++;
        }

        if (i <= len[0] && k <= len[2] && abs(event[0][i]) == 3 && abs(event[2][k]) == 1){
            vector[0][i][0] = vector[0][i - 1][0] + 1;

            vector[2][k][2] = vector[2][k - 1][2] + 1;
            vector[0][i][2] = vector[2][k][2];
            vector[2][k][0] = vector[0][i][0];
            if (event[0][i] < event[2][k]){
                vector[2][k][1] = vector[2][k - 1][1];
                vector[2][k][0] = vector[2][k - 1][0];
                vector[0][i][1] = max(vector[0][i - 1][1], vector[2][k][1]);
            }
            else{
```

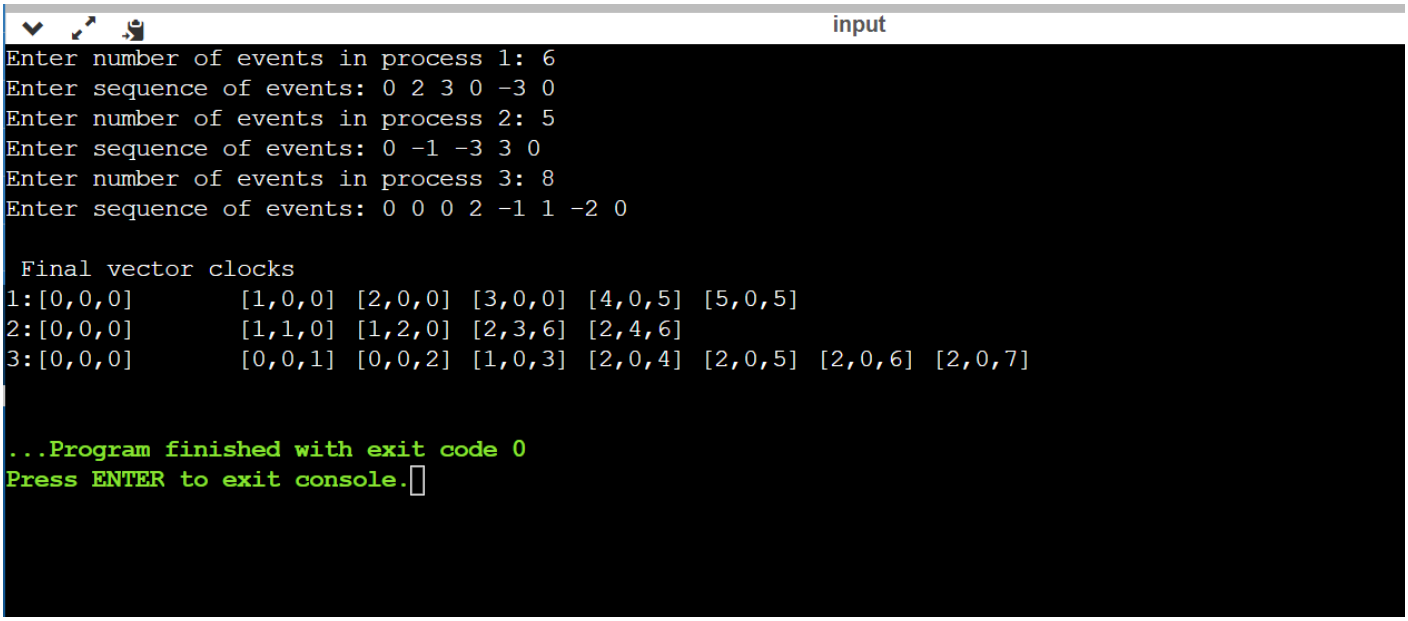
```
                vector[0][i][1] = vector[0][i - 1][1];
                vector[0][i][2] = vector[0][i - 1][2];
                vector[2][k][1] = max(vector[2][k - 1][1], vector[0][i][1]);
            }
            i++;
            k++;
        }
        if (j <= len[1] && k <= len[2] && abs(event[1][j]) == 3 && abs(event[2][k]) == 2){

            vector[1][j][1] = vector[1][j - 1][1] + 1;

            vector[2][k][2] = vector[2][k - 1][2] + 1;
            vector[1][j][2] = vector[2][k][2];
            vector[2][k][1] = vector[1][k][1];
            if (event[1][j] > event[2][k]){
                vector[2][k][0] = vector[2][k - 1][0];
                vector[2][k][1] = vector[2][k - 1][1];

                vector[1][j][0] = max(vector[1][j - 1][0], vector[2][k][0]);
            }
            else{
                vector[1][j][0] = vector[1][j - 1][0];
                vector[1][j][2] = vector[1][j - 1][2];
                vector[2][k][0] = max(vector[2][k - 1][0], vector[1][j][0]);
            }
            j++;
            k++;
        }
    }
    printf("\n Final vector clocks \n");
    for (i = 1; i <= 3; i++){
        printf("%d:", i);
        for (j = 0; j < len[i - 1]; j++){
            printf("[%d,%d,%d]\t", vector[i - 1][j][0], vector[i - 1][j][1], vector[i -
1][j][2]);
        }
        printf("\n");
    }
    return 0;
}
```

Output:


input
Enter number of events in process 1: 6
Enter sequence of events: 0 2 3 0 -3 0
Enter number of events in process 2: 5
Enter sequence of events: 0 -1 -3 3 0
Enter number of events in process 3: 8
Enter sequence of events: 0 0 0 2 -1 1 -2 0
Final vector clocks
1:[0,0,0] [1,0,0] [2,0,0] [3,0,0] [4,0,5] [5,0,5]
2:[0,0,0] [1,1,0] [1,2,0] [2,3,6] [2,4,6]
3:[0,0,0] [0,0,1] [0,0,2] [1,0,3] [2,0,4] [2,0,5] [2,0,6] [2,0,7]
...Program finished with exit code 0
Press ENTER to exit console.