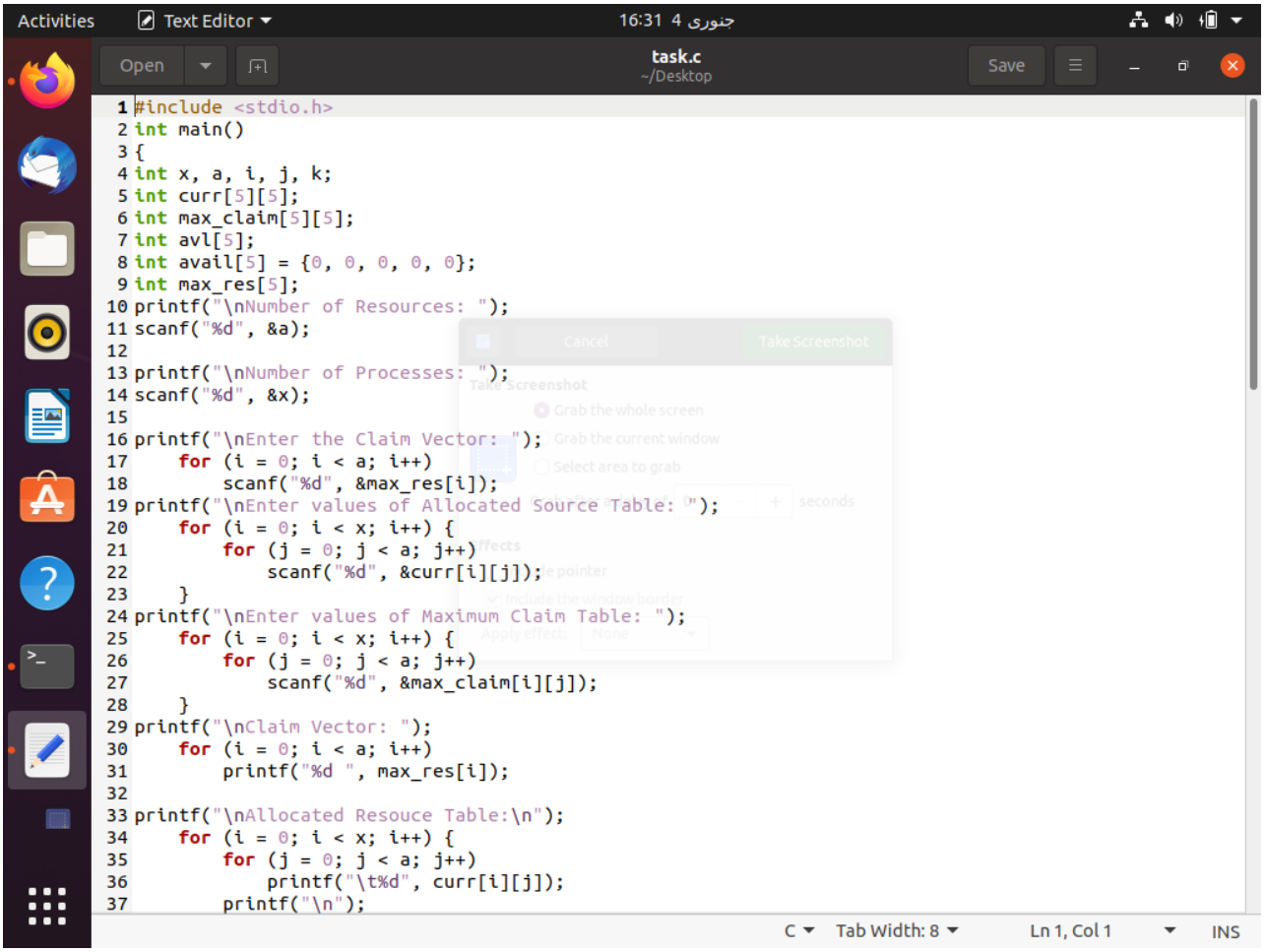
	COLLEGE OF COMPUTING AND INFORMATION SCIENCES		
	Task 13		
Course Title	Operating System	**Total Marks**	05
Date	4-01-21	**Class ID**	108185
Student Id	**11403**	**Student Name**	**Sumaiya Saleh**

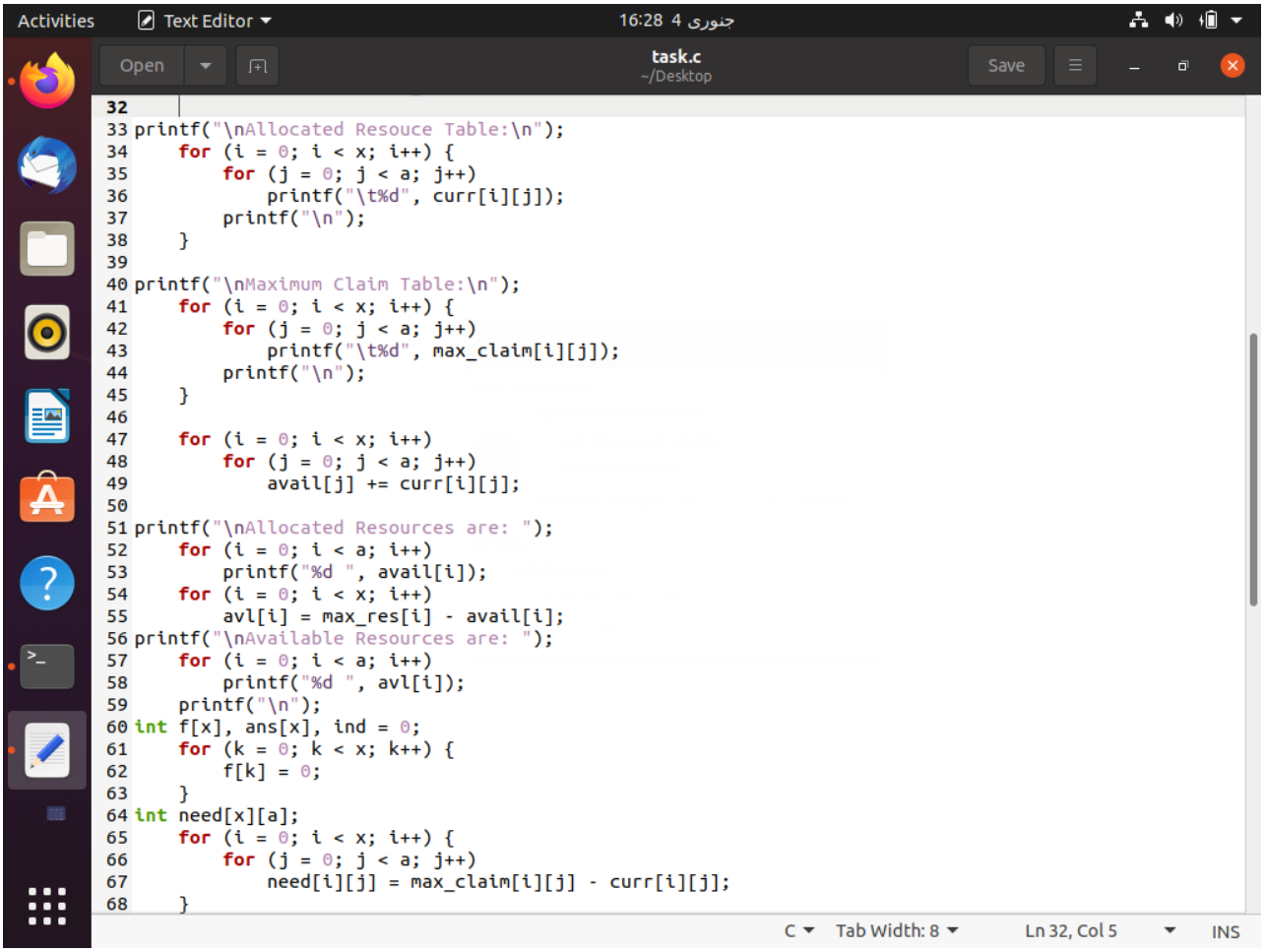
Instructions:

- Copied work and late submission will be marked as ZERO.
- Attach your code and screenshot of your output in this file.
- Submit hardcopy of your solution in class.

Question:

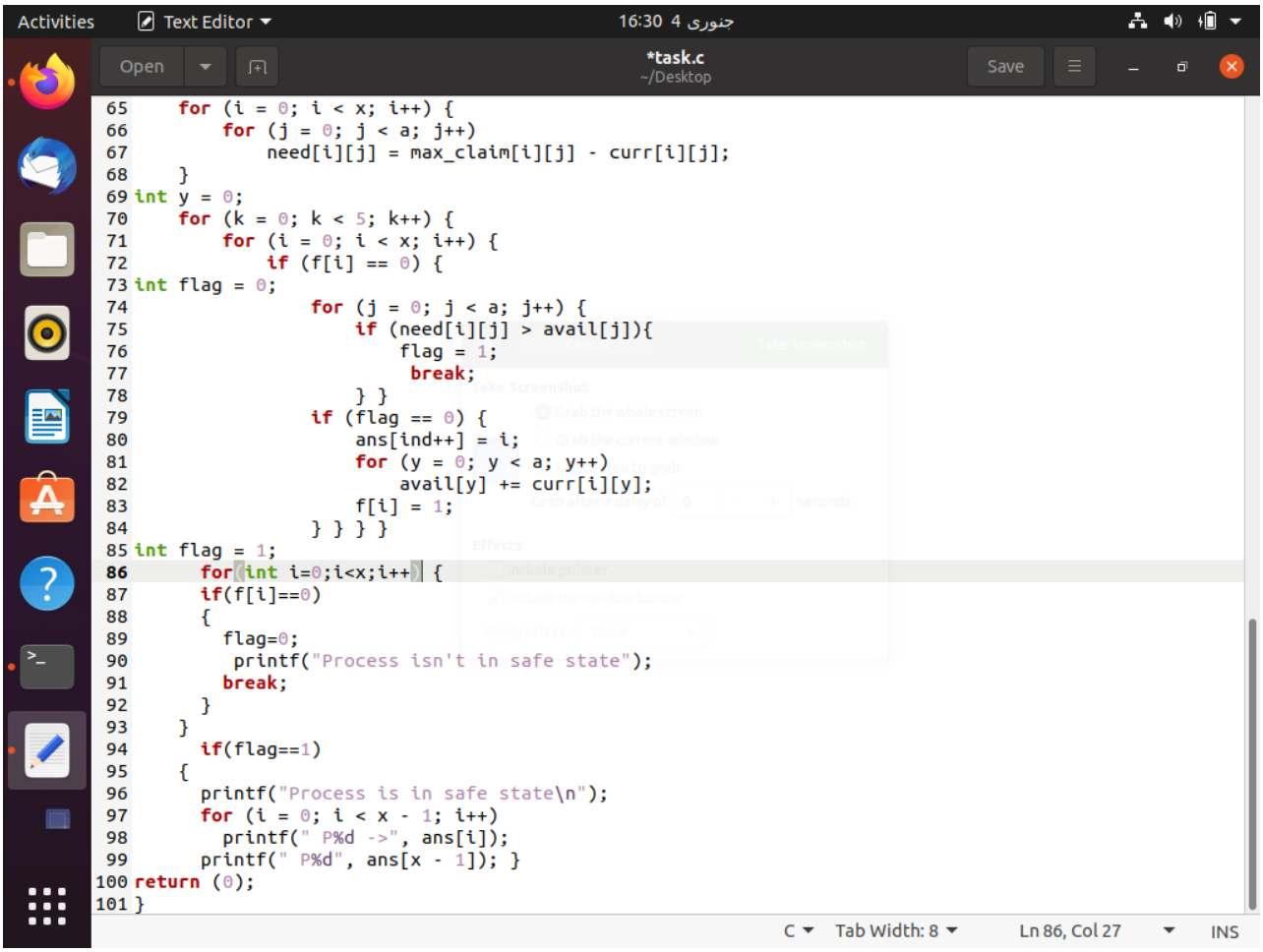
Write a C program for Banker's Algorithm which accepts number of processes, number of resources, Allocation matrix and Maximum Need matrix from user as input. You are asked to calculate the Available Resources matrix and Remaining Need matrix.
In the end print processes, number of resources, Allocation matrix, Maximum Need matrix, Available Resources matrix, Remaining Need matrix and Safe sequence if there is no deadlock.





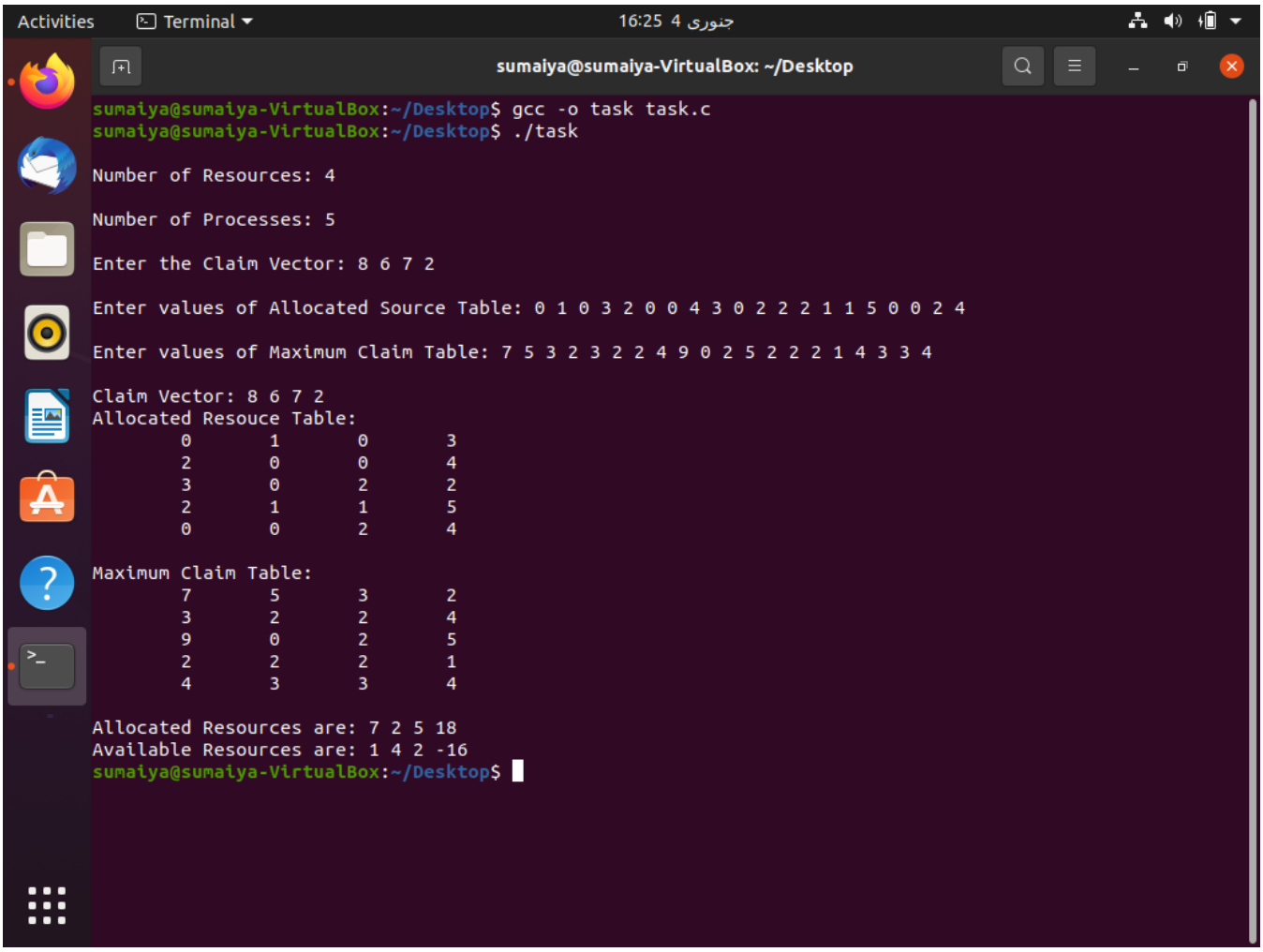

```
32
33 printf("\nAllocated Resouce Table:\n");
34     for (i = 0; i < x; i++) {
35         for (j = 0; j < a; j++)
36             printf("\t%d", curr[i][j]);
37         printf("\n");
38     }
39
40 printf("\nMaximum Claim Table:\n");
41     for (i = 0; i < x; i++) {
42         for (j = 0; j < a; j++)
43             printf("\t%d", max_claim[i][j]);
44         printf("\n");
45     }
46
47     for (i = 0; i < x; i++)
48         for (j = 0; j < a; j++)
49             avail[j] += curr[i][j];
50
51 printf("\nAllocated Resources are: ");
52     for (i = 0; i < a; i++)
53         printf("%d ", avail[i]);
54     for (i = 0; i < x; i++)
55         avl[i] = max_res[i] - avail[i];
56 printf("\nAvailable Resources are: ");
57     for (i = 0; i < a; i++)
58         printf("%d ", avl[i]);
59     printf("\n");
60 int f[x], ans[x], ind = 0;
61     for (k = 0; k < x; k++) {
62         f[k] = 0;
63     }
64 int need[x][a];
65     for (i = 0; i < x; i++) {
66         for (j = 0; j < a; j++)
67             need[i][j] = max_claim[i][j] - curr[i][j];
68     }
```

C Tab Width: 8 Ln 32, Col 5 INS




```
65     for (i = 0; i < x; i++) {
66         for (j = 0; j < a; j++)
67             need[i][j] = max_claim[i][j] - curr[i][j];
68     }
69 int y = 0;
70     for (k = 0; k < 5; k++) {
71         for (i = 0; i < x; i++) {
72             if (f[i] == 0) {
73 int flag = 0;
74                 for (j = 0; j < a; j++) {
75                     if (need[i][j] > avail[j]){
76                         flag = 1;
77                          break;
78                     } }
79                 if (flag == 0) {
80                     ans[ind++] = i;
81                     for (y = 0; y < a; y++)
82                         avail[y] += curr[i][y];
83                     f[i] = 1;
84                 } } } }
85 int flag = 1;
86       for(int i=0;i<x;i++) {
87       if(f[i]==0)
88       {
89         flag=0;
90          printf("Process isn't in safe state");
91         break;
92       }
93     }
94       if(flag==1)
95     {
96       printf("Process is in safe state\n");
97       for (i = 0; i < x - 1; i++)
98         printf(" P%d ->", ans[i]);
99       printf(" P%d", ans[x - 1]); }
100 return (0);
101 }
```

C Tab Width: 8 Ln 86, Col 27 INS


Activities
Terminal
16:25 4 جنوری
sumaiya@sumaiya-VirtualBox: ~/Desktop
sumaiya@sumaiya-VirtualBox:~/Desktop$ gcc -o task task.c
sumaiya@sumaiya-VirtualBox:~/Desktop$ ./task
Number of Resources: 4
Number of Processes: 5
Enter the Claim Vector: 8 6 7 2
Enter values of Allocated Source Table: 0 1 0 3 2 0 0 4 3 0 2 2 2 1 1 5 0 0 2 4
Enter values of Maximum Claim Table: 7 5 3 2 3 2 2 4 9 0 2 5 2 2 2 1 4 3 3 4
Claim Vector: 8 6 7 2
Allocated Resouce Table:
0 1 0 3
2 0 0 4
3 0 2 2
2 1 1 5
0 0 2 4
Maximum Claim Table:
7 5 3 2
3 2 2 4
9 0 2 5
2 2 2 1
4 3 3 4
Allocated Resources are: 7 2 5 18
Available Resources are: 1 4 2 -16
sumaiya@sumaiya-VirtualBox:~/Desktop$