

# LAB REPORT

---

CSE 114 : Data Structure and Algorithms Sessional

PREPARED BY

Mehrin Farzana

ID: 2101013

Session: 2021-2022

Date: 03/08/2023

SUPERVISED BY

Suman Saha

Lecturer

Department of IRE, BDU



BANGABANDHU SHEIKH MUJIBUR

RAHMAN DIGITAL UNIVERSITY

(BDU)

## List of Problems

1. Given the following-

- A knapsack (kind of shoulder bag) with limited weight capacity.
- Few items each having some weight and value.

The problem states,

Which items should be placed into the knapsack such that-

- The value or profit obtained by putting the items into the knapsack is maximum.
- And the weight limit of the knapsack does not exceed.

2. Given n number of activities, their start time and end time, find the maximum number of activities that can be performed without collision.

## Problem No.: 01

### Problem Statement:

Given the following-

- A knapsack (kind of shoulder bag) with limited weight capacity.
- Few items each having some weight and value.

The problem states,

Which items should be placed into the knapsack such that-

- The value or profit obtained by putting the items into the knapsack is maximum.
- And the weight limit of the knapsack does not exceed.

### Code:

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

```
int main()
```

```
{
```

```
    int n;
```

```
    float m, kp=0, kw=0;
```

```
    scanf("%d%f", &n, &m);
```

```
    float p[n], w[n], a[n];
```

```
    for(int i=0; i<n; i++)
```

```
        scanf("%f", &p[i]);
```

```
    for(int i=0; i<n; i++)
```

```
        scanf("%f", &w[i]);
```

```
    for(int i=0; i<n; i++)
```

```
        a[i]=p[i]/w[i];
```

```
    for(int i=0; i<n-1; i++){
```

```
        for(int j=0; j<n-i-1; j++){
```

```
            if(a[j]<a[j+1]){
```

```
                float tmp=p[j];
```

```
                p[j]=p[j+1];
```

```
                p[j+1]=tmp;
```

```
                tmp=w[j];
```

```
                w[j]=w[j+1];
```

```
                w[j+1]=tmp;
```

```
            }
```

```
        }
```

```

    }

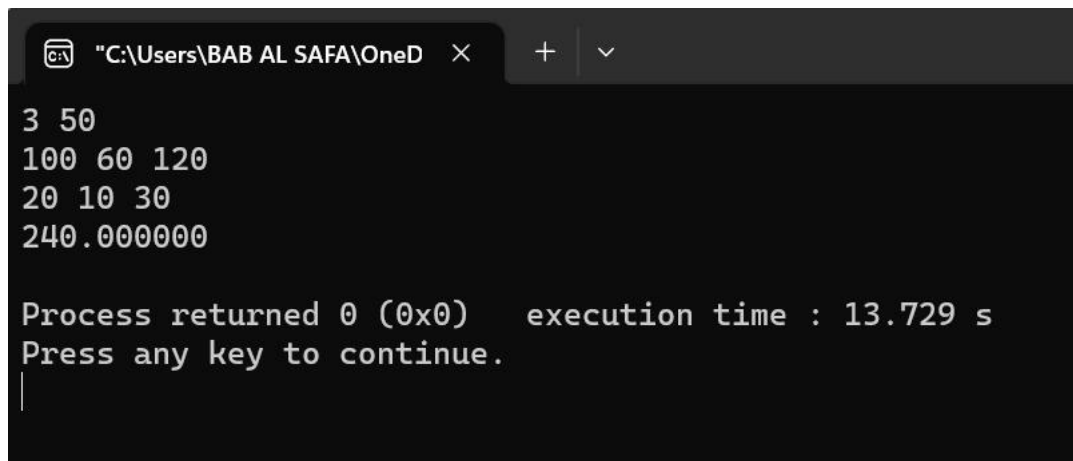
    for(int i=0; i<n; i++){
        if((kw+w[i])>=m){
            kp+=p[i]*((m-kw)/w[i]);
            kw+=((m-kw)/w[i])*w[i];
            break;
        }
        else if((kw+w[i])<m){

            kw+=w[i];
            kp+=p[i];

        }
    }
    printf("%f\n", kp);
    return 0;
}

```

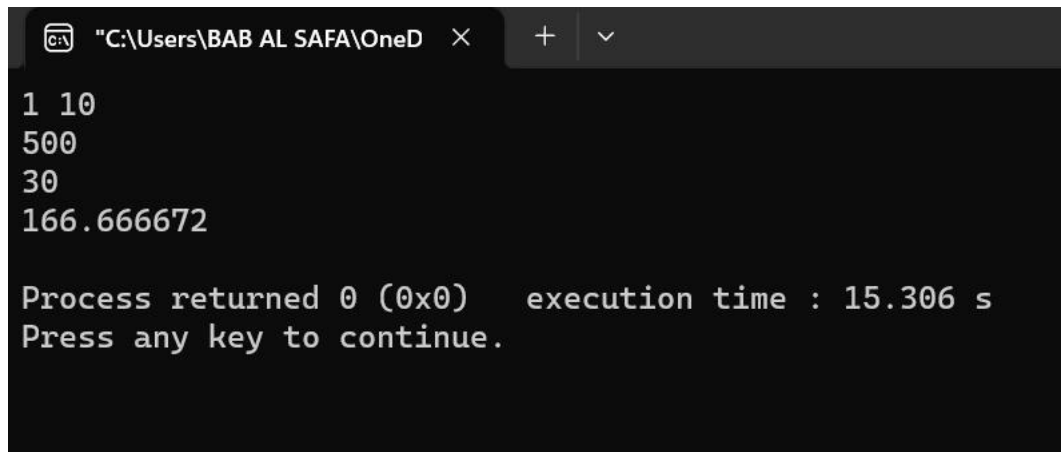
### Output:



```
"C:\Users\BAB AL SAFA\OneD" x + v
3 50
100 60 120
20 10 30
240.000000

Process returned 0 (0x0) execution time : 13.729 s
Press any key to continue.
|
```

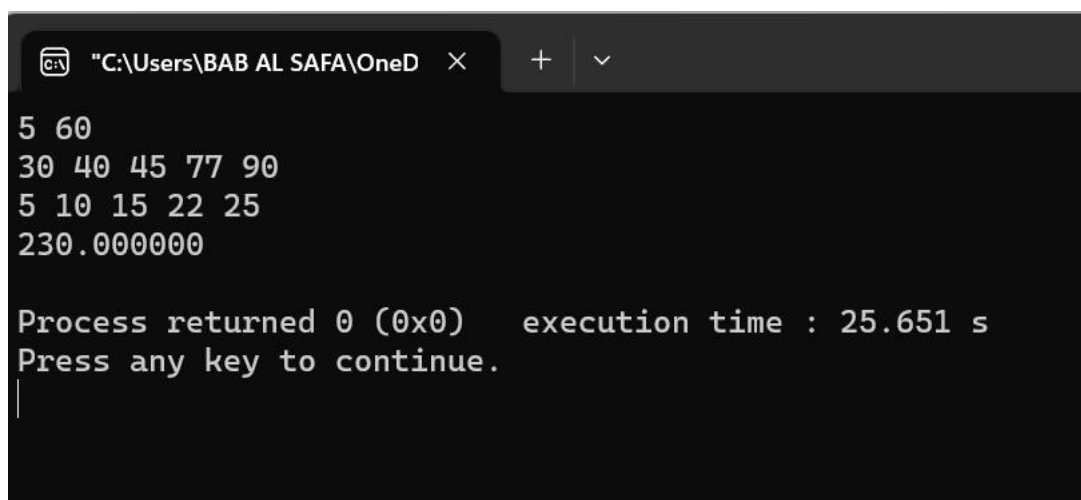
Fig 1.1: Output on console for case 1.



```
"C:\Users\BAB AL SAFA\OneD" x + v
1 10
500
30
166.666672

Process returned 0 (0x0) execution time : 15.306 s
Press any key to continue.
|
```

Fig 1.2: Output on console for case 2.



```
"C:\Users\BAB AL SAFA\OneD" x + v
5 60
30 40 45 77 90
5 10 15 22 25
230.000000

Process returned 0 (0x0) execution time : 25.651 s
Press any key to continue.
|
```

Fig 1.3: Output on console for case 3.

## Problem No.: 02

### Problem Statement:

Given n number of activities, their start time and end time, find the maximum number of activities that can be performed without collision.

### Code:

```
#include <stdio.h>

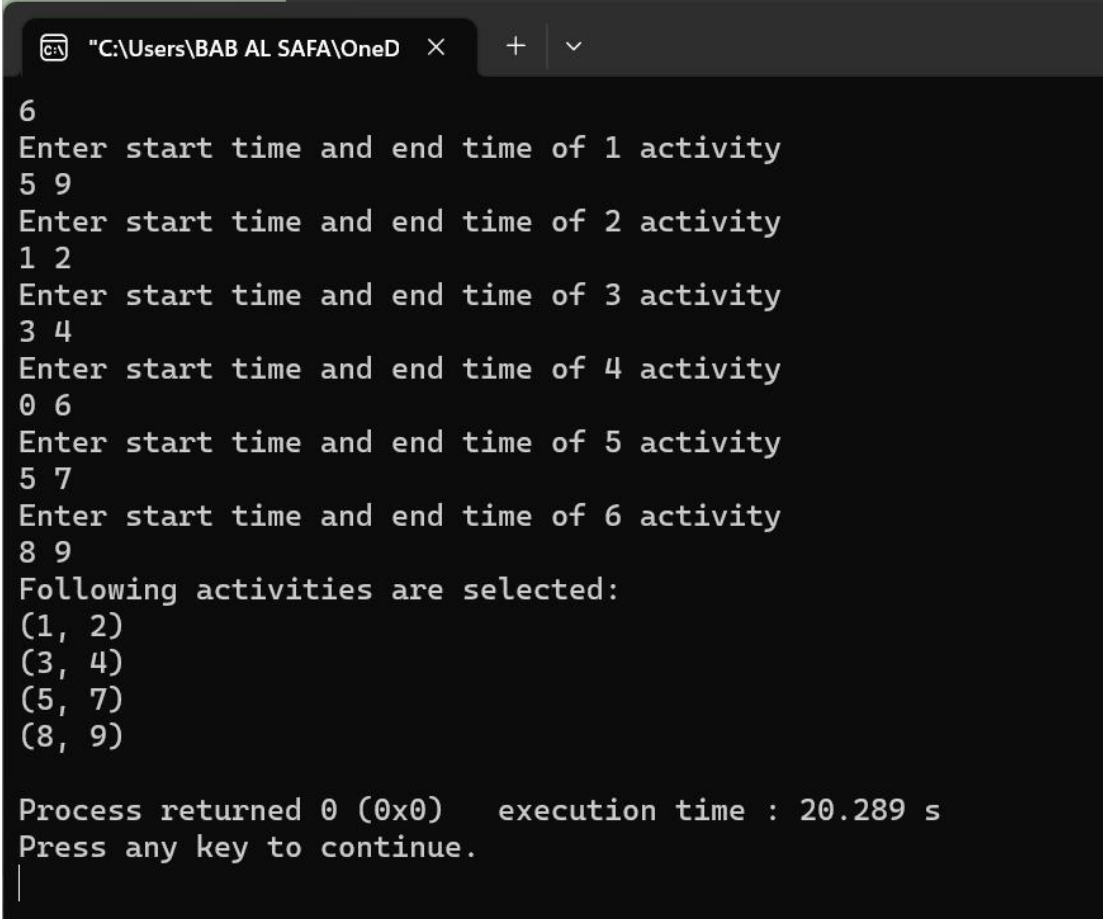
int main()
{
    int n,x;
    scanf("%d", &n);
    int a[n][2];

    for(int i=0; i<n; i++){
        printf("Enter start time and end time of %d activity\n", i+1);
        scanf("%d%d", &a[i][0], &a[i][1]);
    }

    for(int i=0; i<n-1; i++){
        for(int j=0; j<n-i-1; j++){
            if(a[j][1]>a[j+1][1]){
                int tmp = a[j][1];
                a[j][1] = a[j+1][1];
                a[j+1][1] = tmp;

                tmp = a[j][0];
                a[j][0] = a[j+1][0];
                a[j+1][0] = tmp;
            }
        }
    }
    printf("Following activities are selected:\n");
    printf("(%d, %d)\n", a[0][0], a[0][1]);
    x=a[0][1];
    for(int i=1; i<n; i++){
        if(a[i][0]>x){
            x=a[i][1];
            printf("(%d, %d)", a[i][0], a[i][1]);
            printf("\n");
        }
    }
    return 0;
}
```

## Output:



```
"C:\Users\BAB AL SAFA\OneD"
6
Enter start time and end time of 1 activity
5 9
Enter start time and end time of 2 activity
1 2
Enter start time and end time of 3 activity
3 4
Enter start time and end time of 4 activity
0 6
Enter start time and end time of 5 activity
5 7
Enter start time and end time of 6 activity
8 9
Following activities are selected:
(1, 2)
(3, 4)
(5, 7)
(8, 9)

Process returned 0 (0x0)   execution time : 20.289 s
Press any key to continue.
|
```

Fig 2.1: Output on console for case 1.

```
"C:\Users\BAB AL SAFA\OneD  ×  +  ∨  
1 4  
Enter start time and end time of 2 activity  
3 5  
Enter start time and end time of 3 activity  
0 6  
Enter start time and end time of 4 activity  
5 7  
Enter start time and end time of 5 activity  
3 8  
Enter start time and end time of 6 activity  
5 9  
Enter start time and end time of 7 activity  
6 10  
Enter start time and end time of 8 activity  
8 11  
Enter start time and end time of 9 activity  
8 12  
Enter start time and end time of 10 activity  
2 13  
Enter start time and end time of 11 activity  
12 14  
Following activities are selected:  
(1, 4)  
(5, 7)  
(8, 11)  
(12, 14)  
  
Process returned 0 (0x0)   execution time : 38.791 s
```

Fig 2.2: Output on console for case 2.