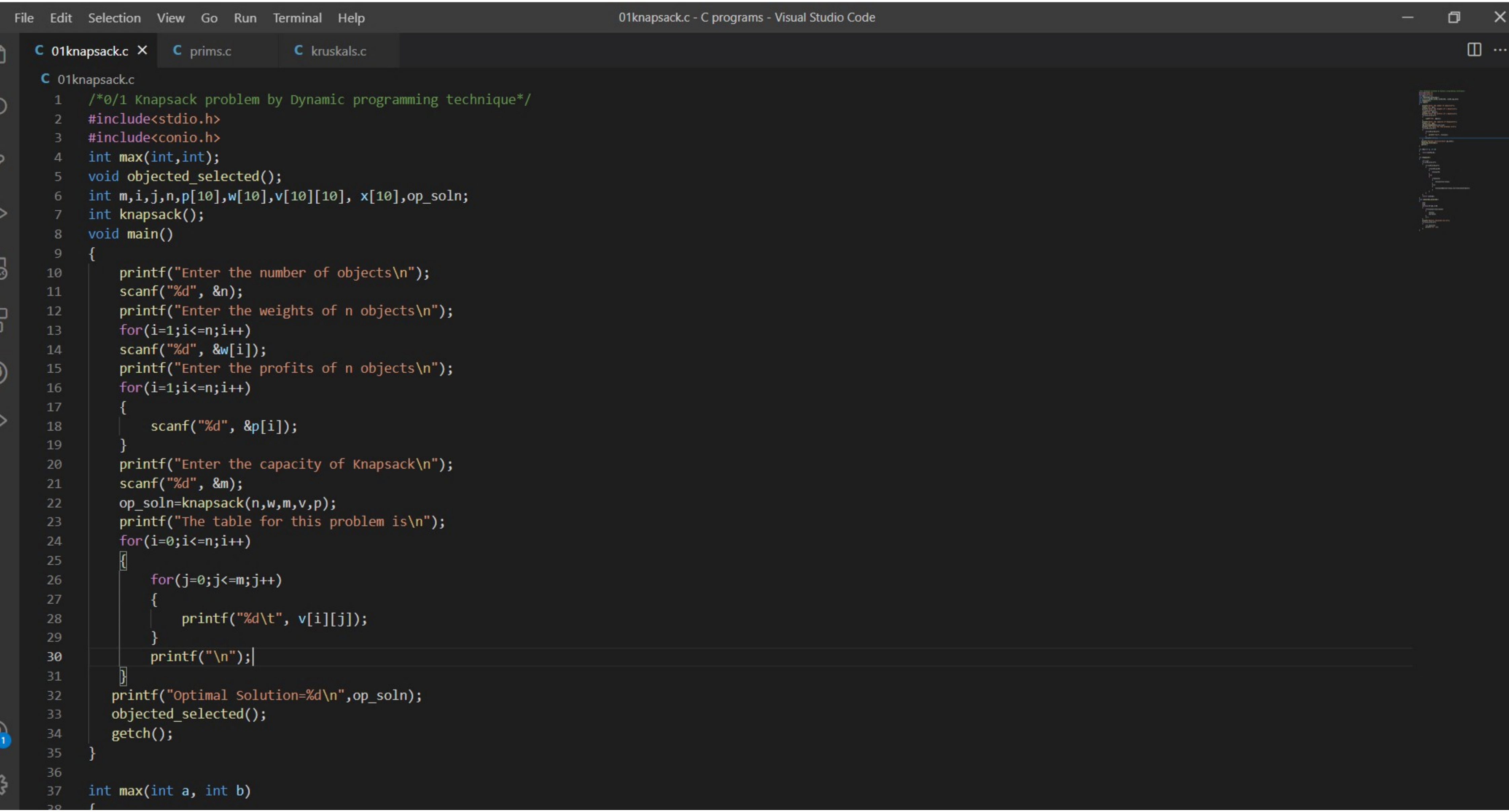
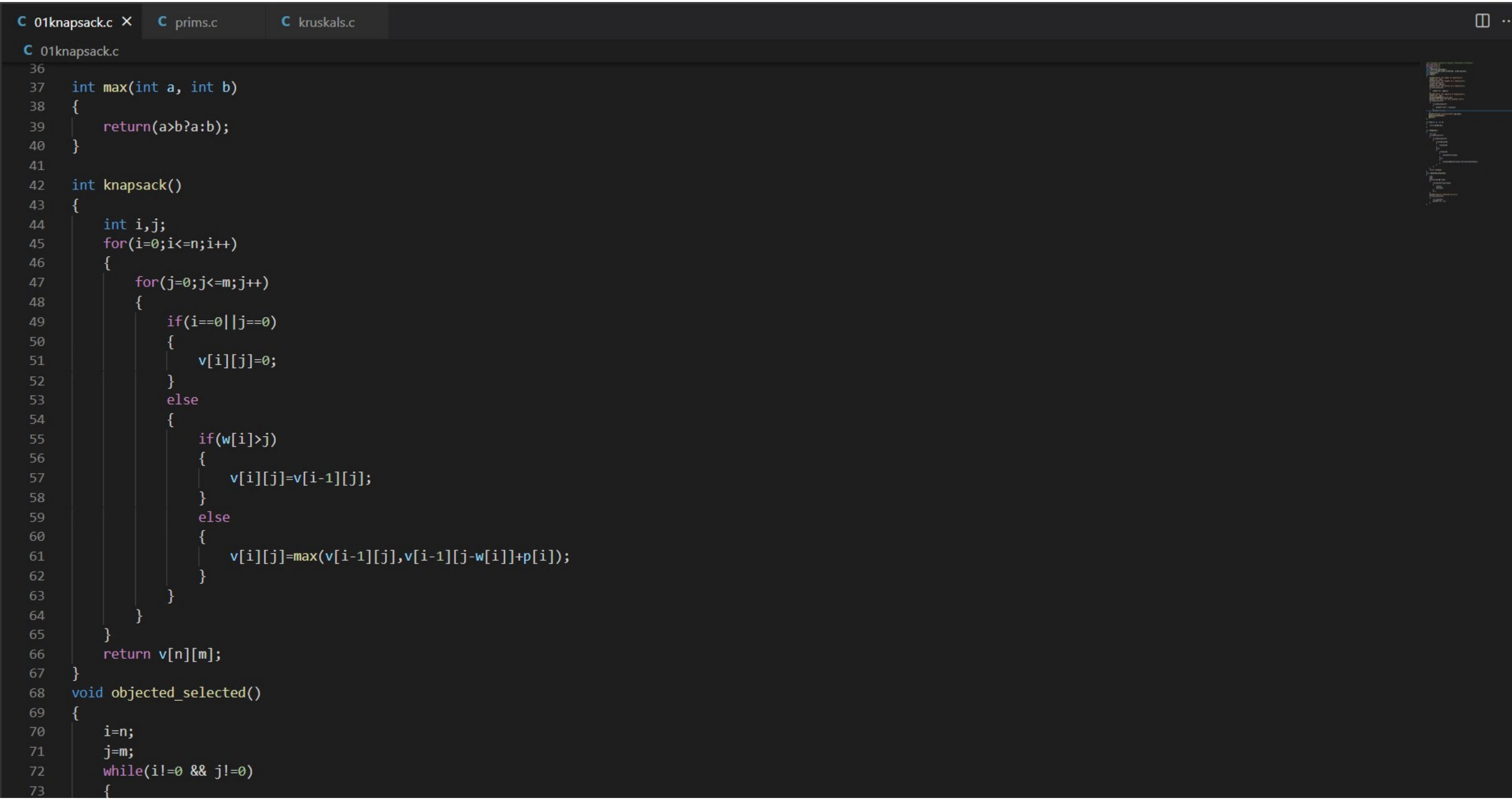
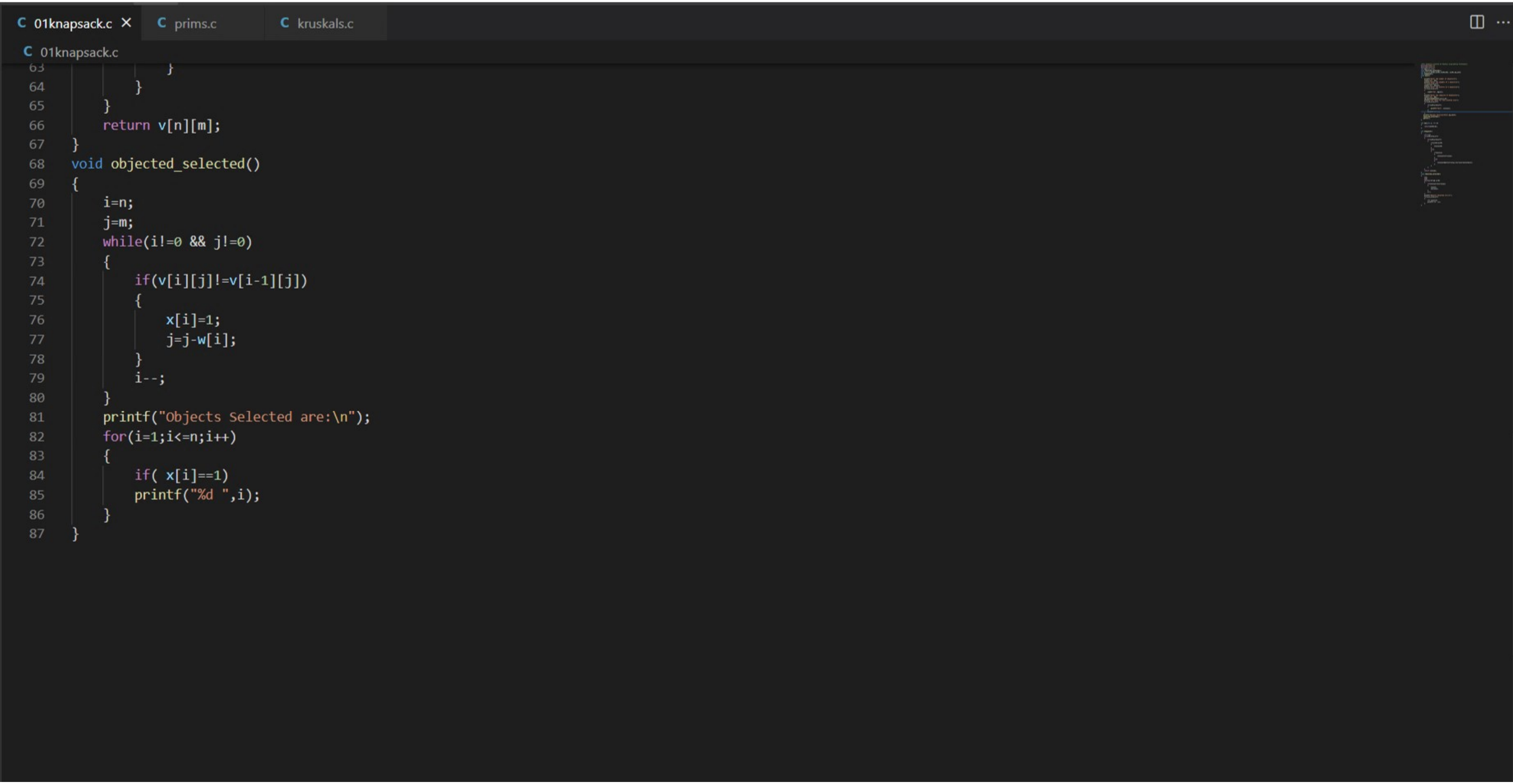


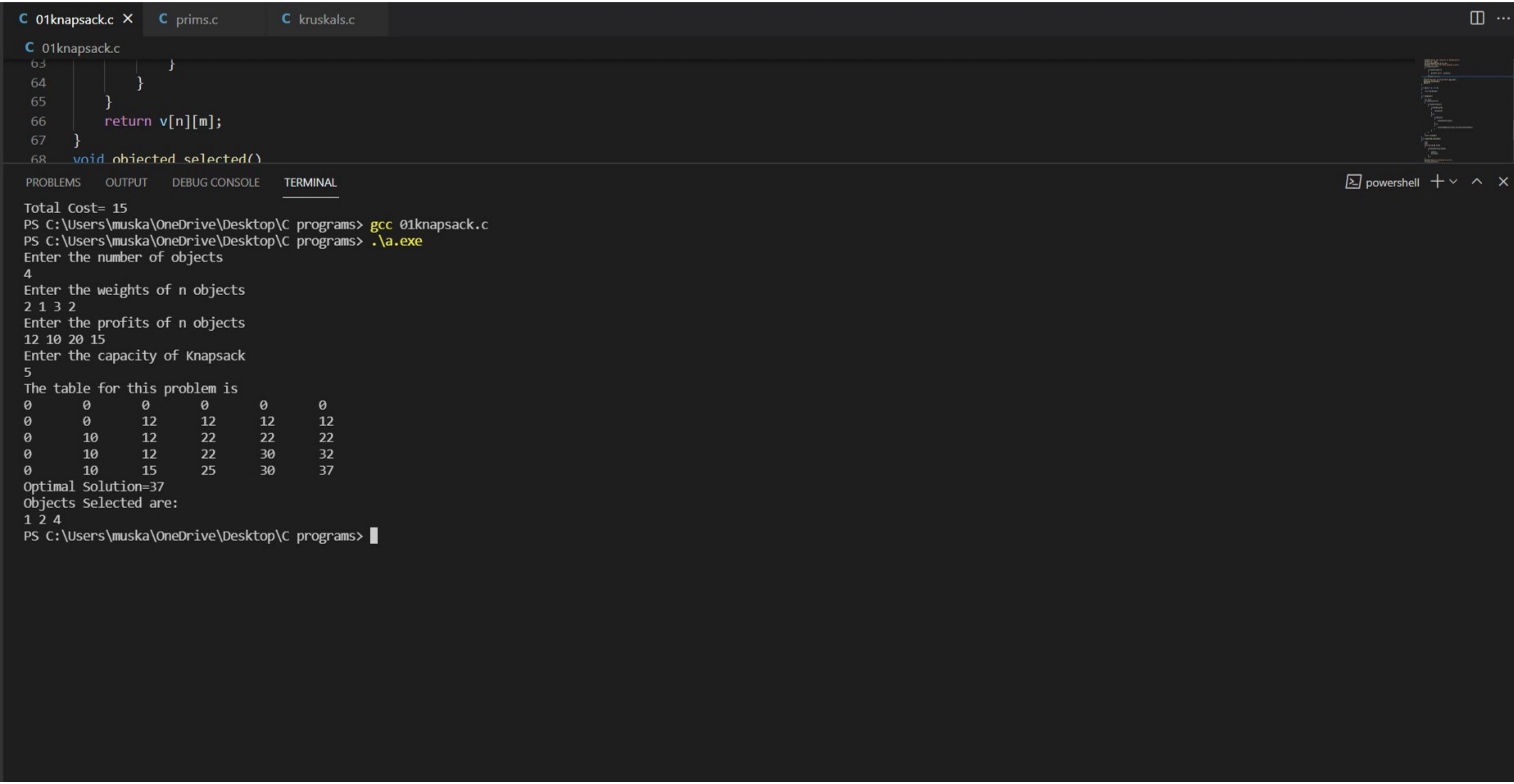
```c
/*0/1 Knapsack problem by Dynamic programming technique*/
#include<stdio.h>
#include<conio.h>
int max(int,int);
void objected_selected();
int m,i,j,n,p[10],w[10],v[10][10], x[10],op_soln;
int knapsack();
void main()
{
    printf("Enter the number of objects\n");
    scanf("%d", &n);
    printf("Enter the weights of n objects\n");
    for(i=1;i<=n;i++)
    scanf("%d", &w[i]);
    printf("Enter the profits of n objects\n");
    for(i=1;i<=n;i++)
    {
        scanf("%d", &p[i]);
    }
    printf("Enter the capacity of Knapsack\n");
    scanf("%d", &m);
    op_soln=knapsack(n,w,m,v,p);
    printf("The table for this problem is\n");
    for(i=0;i<=n;i++)
    {
        for(j=0;j<=m;j++)
        {
            printf("%d\t", v[i][j]);
        }
        printf("\n");
    }
   printf("Optimal Solution=%d\n",op_soln);
   objected_selected();
   getch();
}

int max(int a, int b)
{
```



```c

int max(int a, int b)
{
    return(a>b?a:b);
}

int knapsack()
{
    int i,j;
    for(i=0;i<=n;i++)
    {
        for(j=0;j<=m;j++)
        {
            if(i==0||j==0)
            {
                v[i][j]=0;
            }
            else
            {
                if(w[i]>j)
                {
                    v[i][j]=v[i-1][j];
                }
                else
                {
                    v[i][j]=max(v[i-1][j],v[i-1][j-w[i]]+p[i]);
                }
            }
        }
    }
    return v[n][m];
}
void objected_selected()
{
    i=n;
    j=m;
    while(i!=0 && j!=0)
    {
```



```c
            }
        }
    }
    return v[n][m];
}
void objected_selected()
{
    i=n;
    j=m;
    while(i!=0 && j!=0)
    {
        if(v[i][j]!=v[i-1][j])
        {
            x[i]=1;
            j=j-w[i];
        }
        i--;
    }
    printf("Objects Selected are:\n");
    for(i=1;i<=n;i++)
    {
        if( x[i]==1)
        printf("%d ",i);
    }
}
```



```
            }
        }
    }
    return v[n][m];
}
void objected_selected()
```

```
Total Cost= 15
PS C:\Users\muska\OneDrive\Desktop\C programs> gcc 01knapsack.c
PS C:\Users\muska\OneDrive\Desktop\C programs> .\a.exe
Enter the number of objects
4
Enter the weights of n objects
2 1 3 2
Enter the profits of n objects
12 10 20 15
Enter the capacity of Knapsack
5
The table for this problem is
0       0       0       0       0       0
0       0       12      12      12      12
0       10      12      22      22      22
0       10      12      22      30      32
0       10      15      25      30      37
Optimal Solution=37
Objects Selected are:
1 2 4
PS C:\Users\muska\OneDrive\Desktop\C programs> 
```