

```
/*knapsack*/
```

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

```
int m=0,n=0;
```

```
void knapsack(float cal[],float p[],float w[]);
```

```
int main(){
```

```
    int i;
```

```
    float cal[20],p[20],w[20];
```

```
    printf("enter the max weight of knapsack: ");
```

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

```
    printf("\nenter the no. of objects: ");
```

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

```
    printf("\nenter weights & profits\n");
```

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

```
        printf("w[%d]: ",i+1);
```

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

```
        printf("p[%d]: ",i+1);
```

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

```
    }
```

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

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

```
    }
```

```
    knapsack(cal,p,w);
```

```
    return 0;
```

```
}
```

```
void knapsack(float cal[],float p[],float w[]){
```

```
    int i,j;
```

```
    float temp;
```

```

float total_profit=0;

int selected_items[20];

int ct=0;

for(i=0;i<n;i++){
    for(j=i+1;j<n;j++){
        if(cal[i]<cal[j]){
            temp=cal[i];
            cal[i]=cal[j];
            cal[j]=temp;

            temp=p[i];
            p[i]=p[j];
            p[j]=temp;

            temp=w[i];
            w[i]=w[j];
            w[j]=temp;
        }
    }
}

printf("\n\nprofit\tweight\tcalc\n");

for(i=0;i<n;i++){
    printf("%.3f\t%.3f\t%.3f\n",p[i],w[i],cal[i]);
}

for(i=0;i<n;i++){
    if(m>0 && w[i]<=m){
        m-=w[i];
        total_profit+=p[i];
        selected_items[ct++]=i;
    }
}

```

```

        else{

            break;

        }

    }

// If there is still space in the knapsack, add a fraction of the next item

// if (m > 0 && i < n) {

// float fraction = (float)m / w[i];

// total_profit += p[i] * fraction;

// selected_items[ct] = i;

// Store the index of the selected item

// }

printf("selected items are\n");

for(i=0;i<ct;i++){

    printf("item_id: %d,weight: %.2f,profit:

%.2f\n",selected_items[i]+1,w[selected_items[i]],p[selected_items[i]]);

}

printf("total profit: %.2f\n",total_profit);

}

```

```

C:\Users\HP\OneDrive\Desktop >
enter the max weight of knapsack: 15
enter the no. of objects: 7
enter weights & profits
w[1]: 2
p[1]: 10
w[2]: 3
p[2]: 5
w[3]: 5
p[3]: 15
w[4]: 7
p[4]: 7
w[5]: 1
p[5]: 6
w[6]: 4
p[6]: 18
w[7]: 1
p[7]: 3

profit  weight  calc
6.000   1.000   6.000
10.000  2.000   5.000
18.000  4.000   4.500
15.000  5.000   3.000
3.000   1.000   3.000
5.000   3.000   1.667
7.000   7.000   1.000
selected items are
item_id: 1,weight: 1.00,profit: 6.00
item_id: 2,weight: 2.00,profit: 10.00
item_id: 3,weight: 4.00,profit: 18.00
item_id: 4,weight: 5.00,profit: 15.00
item_id: 5,weight: 1.00,profit: 3.00
total profit: 52.00

-----
Process exited after 49.85 seconds with return value 0
Press any key to continue . . .

```