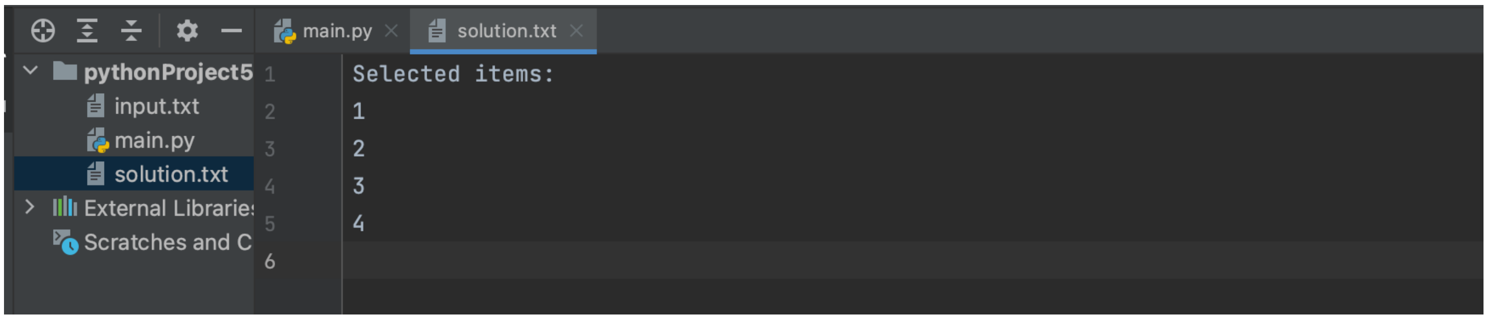
Lab 7
Student name: Khalid Nimri 2140145

Ex1 Code:

```python
def read_input(filename):
    with open(filename, 'r') as file:
        num_items, knapsack_capacity = map(int, file.readline().split())
        items = []
        for _ in range(num_items):
            weight, value = map(int, file.readline().split())
            items.append((weight, value))
    return num_items, knapsack_capacity, items

def write_output(filename, selected_items):
    with open(filename, 'w') as file:
        file.write("Selected items:\n")
        for item in selected_items:
            file.write(f"{item}\n")

def knapsack_dp(num_items, knapsack_capacity, items):
    # Initialize the DP table
    dp_table = [[0] * (knapsack_capacity + 1) for _ in range(num_items + 1)]

    # Construct the DP table
    for i in range(1, num_items + 1):
        weight, value = items[i - 1]
        for w in range(1, knapsack_capacity + 1):
            if weight <= w:
                dp_table[i][w] = max(value + dp_table[i - 1][w - weight], dp_table[i - 
1][w])
            else:
                dp_table[i][w] = dp_table[i - 1][w]

    # Backtracking to determine selected items
    selected_items = []
    w = knapsack_capacity
    for i in range(num_items, 0, -1):
        if dp_table[i][w] != dp_table[i - 1][w]:
            weight, _ = items[i - 1]
            selected_items.append(i)
            w -= weight

    selected_items.reverse()
    return selected_items

def solve_knapsack(input_filename, output_filename):
    num_items, knapsack_capacity, items = read_input(input_filename)
    selected_items = knapsack_dp(num_items, knapsack_capacity, items)
    write_output(output_filename, selected_items)

# Usage
solve_knapsack('input.txt', 'solution.txt')
```

Ex1 Output:



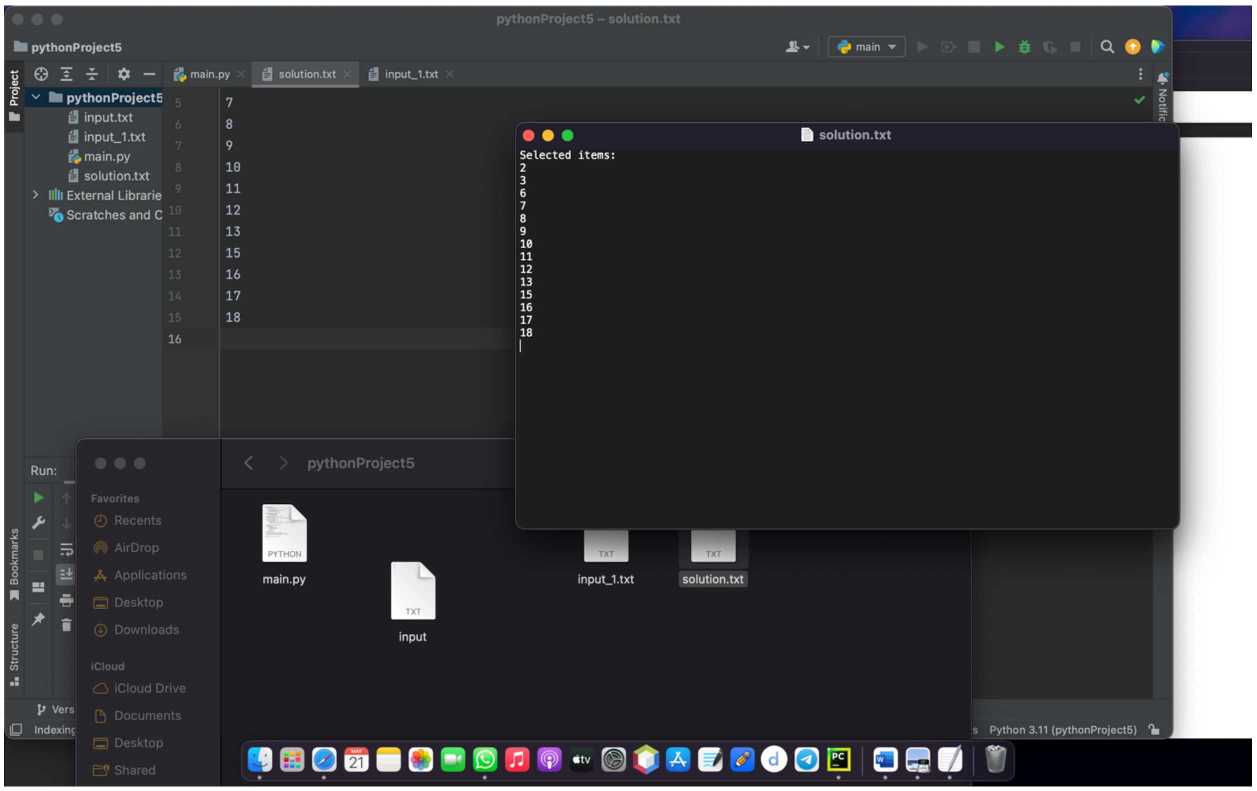
```
Selected items:
1
2
3
4
```

Ex2

Text

```
Selected items:
2
3
6
7
8
9
10
11
12
13
15
16
17
18
```

Screenshot



```
Selected items:
2
3
6
7
8
9
10
11
12
13
15
16
17
18
```