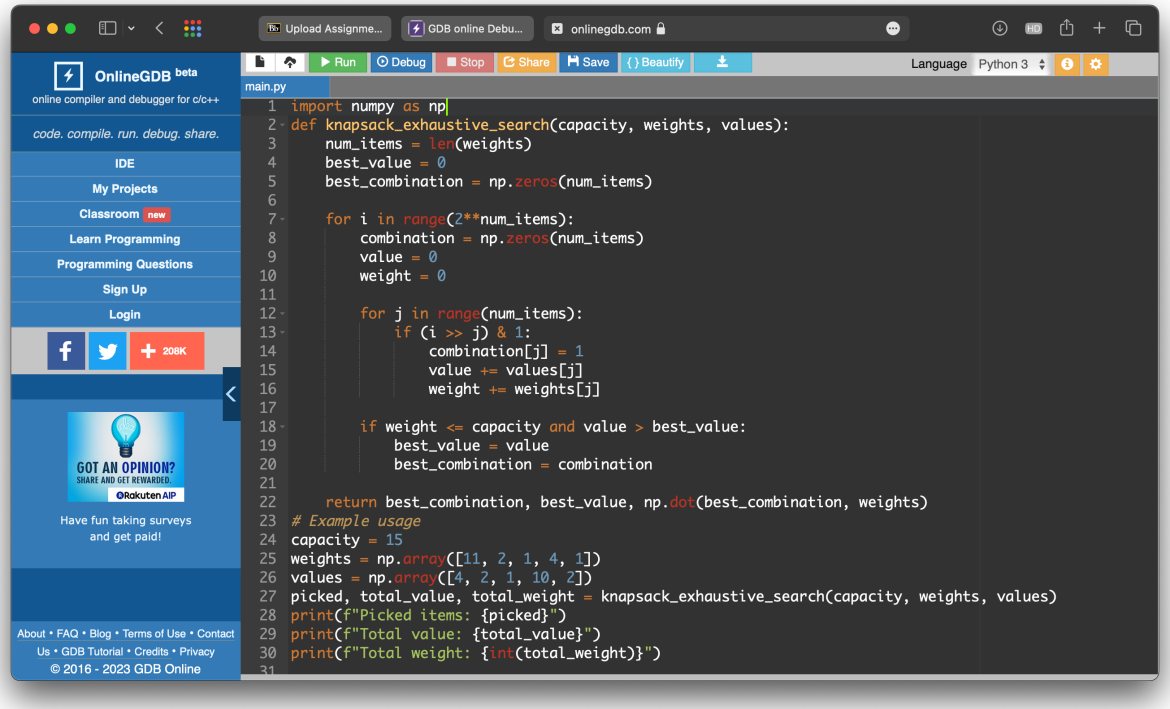
Lab 3
Student name: Khalid Nimri
Student ID: 2140145

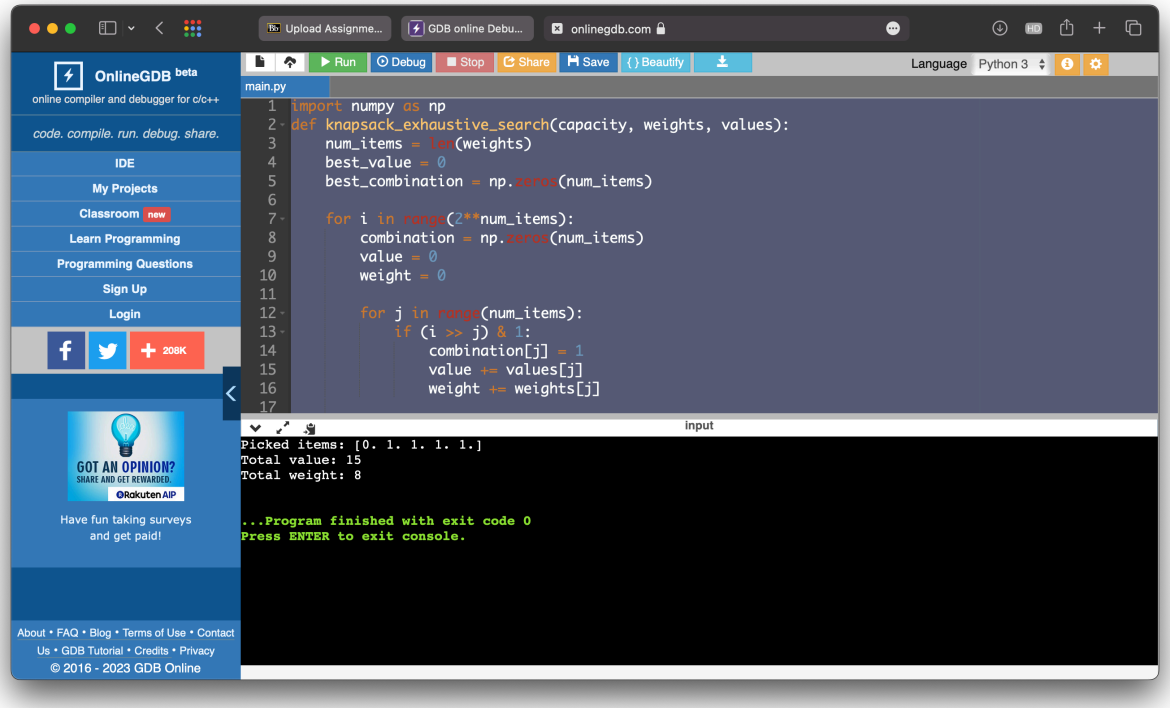
Exercise 1: (Exhaustive Search: Knapsack Problem)




```
import numpy as np
def knapsack_exhaustive_search(capacity, weights, values):
    num_items = len(weights)
    best_value = 0
    best_combination = np.zeros(num_items)

    for i in range(2**num_items):
        combination = np.zeros(num_items)
        value = 0
        weight = 0

        for j in range(num_items):
            if (i >> j) & 1:
                combination[j] = 1
                value += values[j]
                weight += weights[j]
```

```
        if weight <= capacity and value > best_value:
            best_value = value
            best_combination = combination

    return best_combination, best_value, np.dot(best_combination, weights)
# Example usage
capacity = 15
weights = np.array([11, 2, 1, 4, 1])
values = np.array([4, 2, 1, 10, 2])
picked, total_value, total_weight = knapsack_exhaustive_search(capacity, weights, values)
print(f"Picked items: {picked}")
print(f"Total value: {total_value}")
print(f"Total weight: {int(total_weight)}")
```

Output:

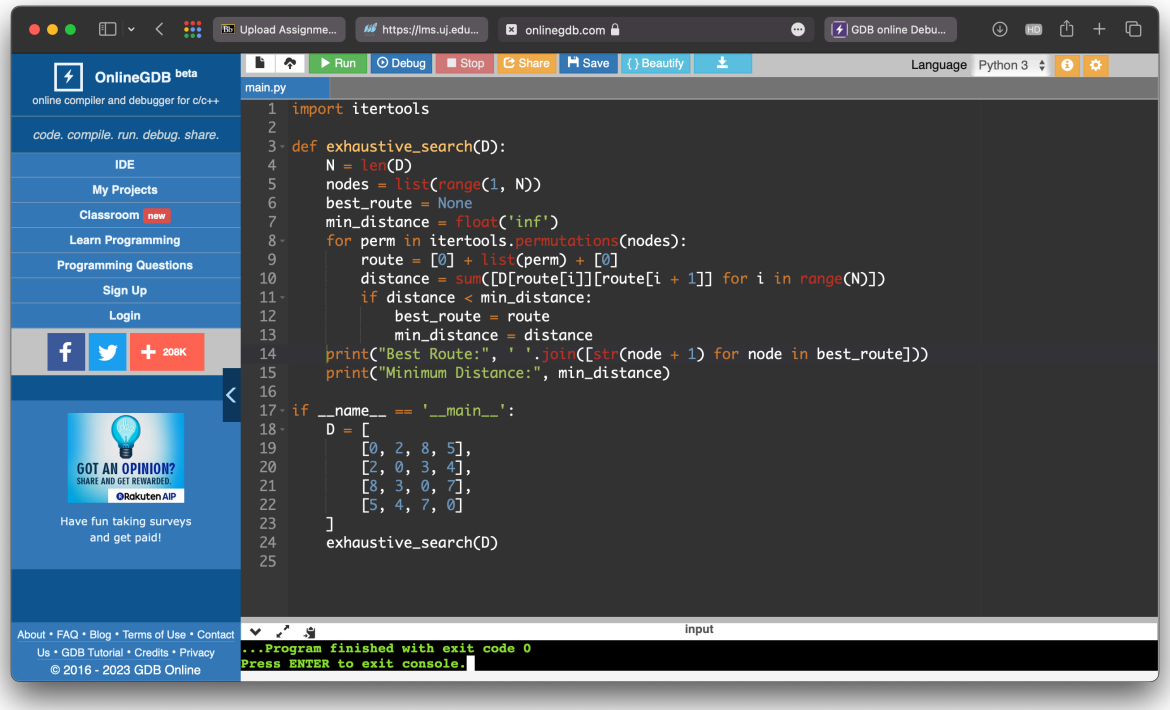


Items Picked P = [0. 1. 1. 1. 1]

Total value of picked items = 15

Total weight of picked items = 8

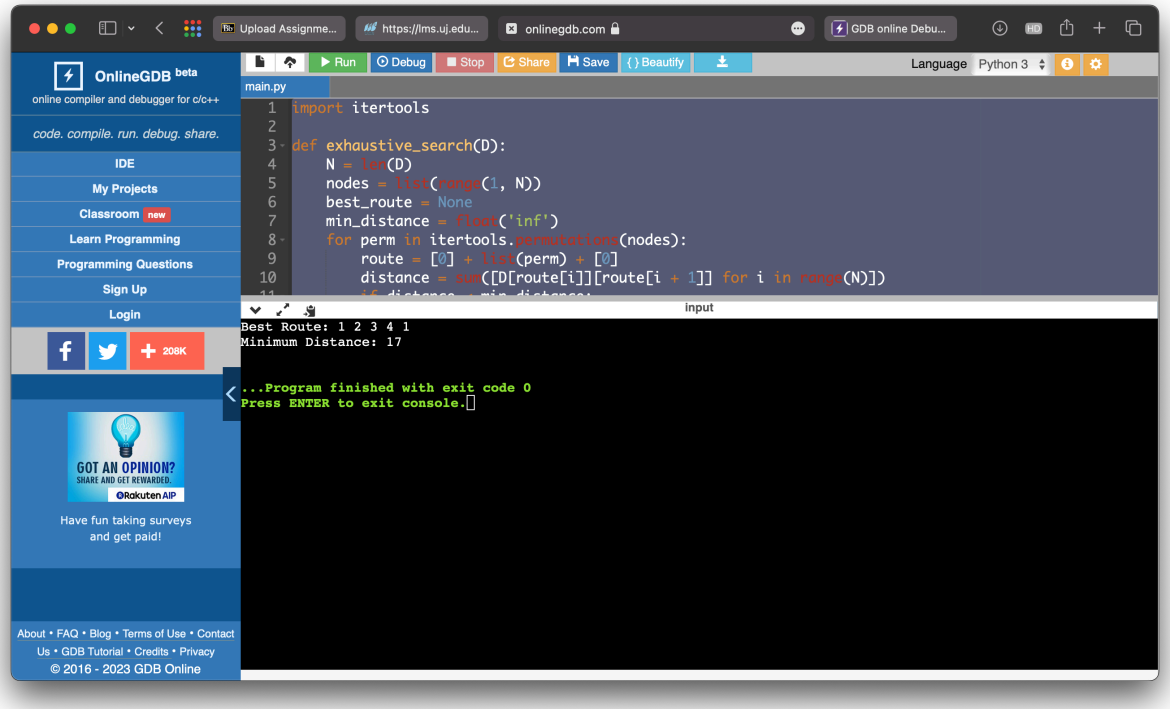
Exercise 2: (Exhaustive Search: Travel Salesperson Problem)




```
import itertools

def exhaustive_search(D):
    N = len(D)
    nodes = list(range(1, N))
    best_route = None
    min_distance = float('inf')
    for perm in itertools.permutations(nodes):
        route = [0] + list(perm) + [0]
        distance = sum([D[route[i]][route[i + 1]] for i in range(N)])
        if distance < min_distance:
            best_route = route
            min_distance = distance
    print("Best Route:", ' '.join([str(node + 1) for node in best_route]))
    print("Minimum Distance:", min_distance)
```

```
if __name__ == '__main__':
    D = [
        [0, 2, 8, 5],
        [2, 0, 3, 4],
        [8, 3, 0, 7],
        [5, 4, 7, 0]
    ]
    exhaustive_search(D)
```

Output:



the best path: 1 2 3 4 1

minimum distance: 17