Programiz
Python Online Compiler

Ad

KTM DUKE WITH COLOR TFT
KTM - READY TO RACE
Get a call back

Programiz PRO ›

main.py

Share

Run

Output

Clear

```python
def maxCoins(piles):
    piles.sort()
    total_coins = 0
    n = len(piles) // 3
    for i in range(n):
        total_coins += piles[-(2 + i)]
    return total_coins
print(maxCoins([2, 4, 1, 2, 7, 8]))
print(maxCoins([2, 4, 5]))
```

```
11
4

=== Code Execution Successful ===
```

SH113 / Dr MGR...
Closed road
Search
ENG
IN
5:12 PM
10/20/2024

Programiz
Python Online Compiler

Programiz PRO ›

main.py | Share | Run

```python
def min_coins_to_add(coins, target):
    coins.sort()
    current_sum = 0
    coins_needed = 0
    for coin in coins:
        while current_sum + 1 < coin and current_sum < target:
            current_sum += current_sum + 1
            coins_needed += 1
        current_sum += coin
    while current_sum < target:
        current_sum += current_sum + 1
        coins_needed += 1
    return coins_needed
coins = [1, 4, 10]
target = 19
print(min_coins_to_add(coins, target))
```

Output | Clear

```
2

=== Code Execution Successful ===
```

SH113 / Dr MGR...
Closed road
Search
ENG
IN
5:12 PM
10/20/2024

Programiz
Python Online Compiler

Programiz PRO ›

main.py | Share | Run

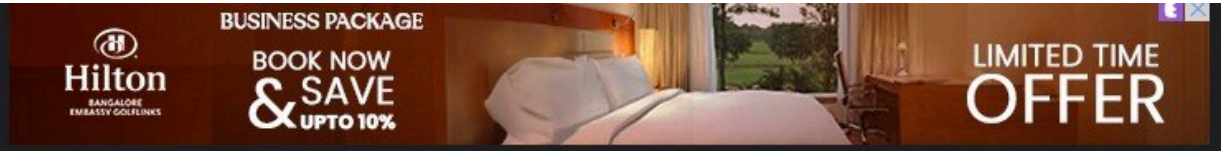
```python
def minimumTimeRequired(jobs, k):
    def canDistribute(maxTime):
        workers = [0] * k
        return backtrack(0, workers, maxTime)
    def backtrack(i, workers, maxTime):
        if i == len(jobs):
            return True
        for j in range(k):
            if workers[j] + jobs[i] <= maxTime:
                workers[j] += jobs[i]
                if backtrack(i + 1, workers, maxTime):
                    return True
                workers[j] -= jobs[i]
            if workers[j] == 0:
                break
        return False
    left, right = max(jobs), sum(jobs)
    while left < right:
        mid = (left + right) // 2
        if canDistribute(mid):
            right = mid
        else:
```

Output | Clear

```
3

=== Code Execution Successful ===
```

Hot weather
Now
Search
ENG
IN
5:14 PM
10/20/2024

Programiz
Python Online Compiler



Programiz PRO ›

main.py

Share Run

```python
def min_containers(weights, max_capacity):
    weights.sort(reverse=True)
    containers = 0
    current_capacity = 0

    for weight in weights:
        if current_capacity + weight > max_capacity:
            containers += 1
            current_capacity = weight
        else:
            current_capacity += weight

    if current_capacity > 0:
        containers += 1

    return containers
n = 7
weights = [5, 10, 15, 20, 25, 30, 35]
max_capacity = 50
output = min_containers(weights, max_capacity)
print(output)

```

Output Clear

```
4

=== Code Execution Successful ===
```

Sunset coming
6:17 PM
Search
ENG
IN
5:21 PM
10/20/2024

Programiz
Python Online Compiler

KTM DUKE WITH COLOR TFT
KTM - READY TO RACE
Get a call back

Programiz PRO >

main.py | Share | Run

```python
import numpy as np
def dijkstra(graph, source):
    n = len(graph)
    distances = [float('inf')] * n
    distances[source] = 0
    visited = [False] * n
    for _ in range(n):
        min_distance = float('inf')
        min_index = -1
        for v in range(n):
            if not visited[v] and distances[v] < min_distance:
                min_distance = distances[v]
                min_index = v
        visited[min_index] = True
        for v in range(n):
            if (graph[min_index][v] != float('inf') and
                not visited[v] and
                distances[min_index] + graph[min_index][v] <
                    distances[v]):
                distances[v] = distances[min_index] +
                    graph[min_index][v]
    return distances
```

Output | Clear

```
[0, 7, 3, 9, 5]

=== Code Execution Successful ===
```

Hot weather
Now
Search
ENG
IN
5:15 PM
10/20/2024

Programiz
Python Online Compiler

Ad

KTM DUKE WITH COLOR TFT
KTM - READY TO RACE
Get a call back

Programiz PRO ›

main.py | Share | Run | Output | Clear

```python
import heapq

def dijkstra(n, edges, source, target):
    graph = {i: [] for i in range(n)}
    for u, v, w in edges:
        graph[u].append((v, w))
        graph[v].append((u, w))

    min_heap = [(0, source)]
    distances = {i: float('inf') for i in range(n)}
    distances[source] = 0

    while min_heap:
        current_distance, current_vertex = heapq.heappop(min_heap)

        if current_distance > distances[current_vertex]:
            continue

        for neighbor, weight in graph[current_vertex]:
            distance = current_distance + weight

            if distance < distances[neighbor]:
```

```
20

=== Code Execution Successful ===
```

PHI - NYG
In 6 hours
Search
ENG
IN
5:17 PM
10/20/2024

Programiz
Python Online Compiler

Programiz PRO ›

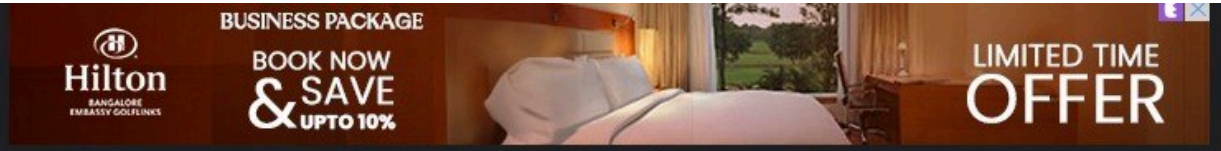
main.py | Share | Run | Output | Clear

```python
def max_weight(weights, max_capacity):
    weights.sort(reverse=True)
    total_weight = 0

    for weight in weights:
        if total_weight + weight <= max_capacity:
            total_weight += weight

    return total_weight
n = 5
weights = [10, 20, 30, 40, 50]
max_capacity = 60
output = max_weight(weights, max_capacity)
print(output)

```

```
60

=== Code Execution Successful ===
```

Sunset coming
6:17 PM
Search
ENG
IN
5:20 PM
10/20/2024

Programiz
Python Online Compiler



Programiz PRO ›

main.py

Share Run

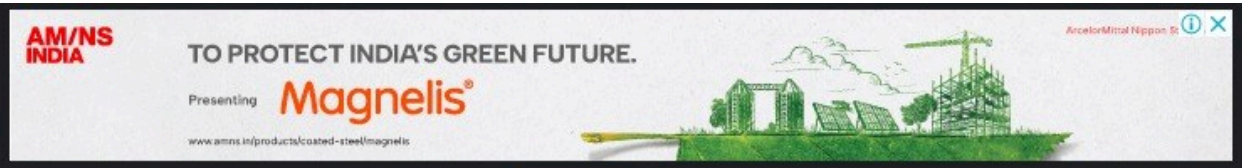
```python
def min_containers(weights, max_capacity):
    weights.sort(reverse=True)
    containers = 0
    current_capacity = 0

    for weight in weights:
        if current_capacity + weight > max_capacity:
            containers += 1
            current_capacity = weight
        else:
            current_capacity += weight

    if current_capacity > 0:
        containers += 1

    return containers
n = 7
weights = [5, 10, 15, 20, 25, 30, 35]
max_capacity = 50
output = min_containers(weights, max_capacity)
print(output)

```

Output Clear

```
4

=== Code Execution Successful ===
```

Sunset coming
6:17 PM
Search
ENG
IN
5:21 PM
10/20/2024

Programiz
Python Online Compiler



Programiz PRO ›

main.py

Share | Run

```python
class DisjointSet:
    def __init__(self, n):
        self.parent = list(range(n))
        self.rank = [0] * n

    def find(self, u):
        if self.parent[u] != u:
            self.parent[u] = self.find(self.parent[u])
        return self.parent[u]

    def union(self, u, v):
        root_u = self.find(u)
        root_v = self.find(v)
        if root_u != root_v:
            if self.rank[root_u] > self.rank[root_v]:
                self.parent[root_v] = root_u
            elif self.rank[root_u] < self.rank[root_v]:
                self.parent[root_u] = root_v
            else:
                self.parent[root_v] = root_u
                self.rank[root_u] += 1

```

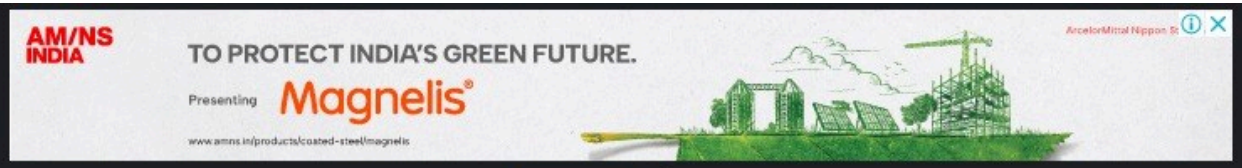
Output

Clear

```
Edges in MST: [(2, 3, 4), (0, 3, 5), (0, 1, 10)]
Total weight of MST: 19

=== Code Execution Successful ===
```

86°F
Partly sunny
Search
ENG
IN
5:22 PM
10/20/2024

Programiz
Python Online Compiler



Programiz PRO ›

main.py

Share | Run

```python
class DisjointSet:
    def __init__(self, n):
        self.parent = list(range(n))
        self.rank = [0] * n

    def find(self, u):
        if self.parent[u] != u:
            self.parent[u] = self.find(self.parent[u])
        return self.parent[u]

    def union(self, u, v):
        root_u = self.find(u)
        root_v = self.find(v)
        if root_u != root_v:
            if self.rank[root_u] > self.rank[root_v]:
                self.parent[root_v] = root_u
            elif self.rank[root_u] < self.rank[root_v]:
                self.parent[root_u] = root_v
            else:
                self.parent[root_v] = root_u
                self.rank[root_u] += 1

```

Output

Clear

```
Edges in MST: [(2, 3, 4), (0, 3, 5), (0, 1, 10)]
Total weight of MST: 19

=== Code Execution Successful ===
```

86°F
Partly sunny
Search
ENG
IN
5:22 PM
10/20/2024

Programiz
Python Online Compiler

Programiz PRO ›

main.py | Share | Run

```python
def find_mst(n, edges):
    parent = list(range(n))
    rank = [0] * n

    def find(x):
        if parent[x] != x:
            parent[x] = find(parent[x])
        return parent[x]

    def union(x, y):
        rootX = find(x)
        rootY = find(y)
        if rootX != rootY:
            if rank[rootX] > rank[rootY]:
                parent[rootY] = rootX
            elif rank[rootX] < rank[rootY]:
                parent[rootX] = rootY
            else:
                parent[rootY] = rootX
                rank[rootX] += 1

    edges.sort(key=lambda x: x[2])
```

Output | Clear

```
Is the given MST unique? True

=== Code Execution Successful ===
```

86°F
Partly sunny
Search
ENG
IN
5:23 PM
10/20/2024