UEH
UNIVERSITY

ÁP DỤNG GIẢI THUẬT DI TRUYỀN CHO BÀI TOÁN KNAPSACK

Giảng viên: TS. Đặng Ngọc Hoàng Thành
Mã lớp học phần: 23D1INF50904203

UEH
UNIVERSITY

NỘI DUNG

1. Tổng quan đề tài

- Bài toán Knapsack
- Phát biểu bài toán
- Một số hướng tiếp cận
- So sánh độ phức tạp

2.Giải thuật di truyền

- Khái niệm
- Ưu điểm
- Cài đặt thuật toán

3.Kết quả thực nghiệm

- Một số tình huống đạt được

4. Giao diện

- Trực quan giao diện người dùng

5.Ứng dụng thực tiễn

- Tối ưu hoá kế hoạch sản xuất
- Tối ưu hoá đường vận chuyển

UEH
UNIVERSITY

TỔNG QUAN ĐỀ TÀI

UEH
UNIVERSITY

BÀI TOÁN KNAPSACK

UEH UNIVERSITY

Một túi có **giới hạn dung lượng** và một tập hợp các đồ vật, mỗi đồ vật có giá trị và khối lượng khác nhau.

Mục tiêu của bài toán là đưa các đồ vật vào túi sao cho:

Tổng giá trị của các đồ vật này là **lớn nhất**

Tổng khối lượng của các đồ vật **không vượt quá** khối lượng giới hạn của túi.

UEH UNIVERSITY

Ta phát biểu bài toán như sau:

- Cho n là số vật phẩm
- w_i là cân nặng của vật thứ i $(1 \leq i \leq n)$
- v_i là giá trị của vật thứ i $(1 \leq i \leq n)$
- W là khả năng chứa tối đa của ba lô.

$$max \sum_{i=1}^{n} v_i x_i$$

với $\sum_{i=1}^{n} w_i x_i \leq W$ và $x_i \in \{0, 1\}$, $1 \leq i \leq n$

UEH UNIVERSITY

MỘT SỐ HƯỚNG TIẾP CẬN

Brute-Force	**Liệt kê** tất cả các tập con của tập hợp đồ vật, kiểm tra **Kiểm tra** xem tập con nào có tổng trọng lượng không vượt quá giới hạn trọng lượng của túi
Greedy Algorithm	**Sắp xếp** các đồ vật theo thứ tự giá trị giảm dần hoặc trọng lượng tăng dần. **Chọn** đồ vật đầu tiên và thêm vào túi, tiếp tục chọn đồ vật thứ hai, thứ ba, cho đến khi đạt giới hạn trọng lượng của túi hoặc hết các đồ vật.

UEH UNIVERSITY

SO SÁNH ĐỘ PHỨC TẠP

Greedy Algorithm

$O(n^2) \rightarrow O(nlogn)$

Genetic Algorithm

$O(n \times g \times c)$

n là kích thước của không gian tìm kiếm
g là số lượng thế hệ
c là số lượng cá thể trong mỗi thế hệ

Brute-Force

$O(2^n)$

UEH
UNIVERSITY

GIẢI THUẬT DI TRUYỀN

Khái niệm và kỹ thuật liên quan đến di truyền học

Các sinh vật → Di truyền Gen / Chọn lọc tự nhiên → **Cá thể mạnh mẽ và phù hợp**

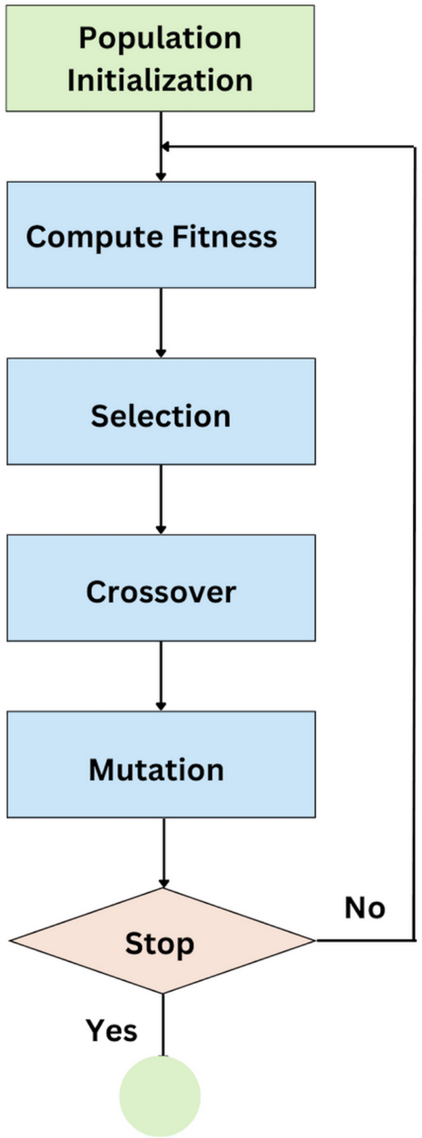
UEH UNIVERSITY

Ưu điểm của GTDT

Giải thuật di truyền có khả năng tìm được các giải pháp tối ưu cho:

- Các không gian tìm kiếm phức tạp và không liên tục
- Các bài toán tối ưu hóa đa mục tiêu, trong đó có nhiều hàm mục tiêu cần được tối ưu hóa đồng thời

Lưu đồ Giải thuật di truyền



UEH UNIVERSITY

PSEUDOCODE

Algorithm 1 Genetic Algorithm

GA(Fitness,$\theta, n, r_{co}, r_{mu}$)

Data:

Fitness: A function that produces the score (fitness) given a hypothesis

θ: The desired fitness value (i.e., a threshold specifying the termination condition)

n: The number of hypotheses in the population

r_{co}: The percentage of the population influenced by the crossover operator at each step

r_{mu}: The percentage of the population influenced by the mutation operator at each step

Result: Optimal solution

Initialize the population: H Randomly generate n hypotheses;

Evaluate the initial population. For each $h \in H$ compute Fitness(h);

while $max_{\{h \in H\}} Fitness(h) < \theta$ **do**

  H^{next};

  Reproduction

  Probabilistically select $(1 - r_{co})n$ hypotheses of H to add to H^{next};

  The probability of selecting hypothesis h_i from H is $P(h_i) = \frac{Fitness(h_i)}{\sum_{j=1}^{n} Fitness(h_j)}$;

  Crossover

  Probabilistically select $(r_{co}.n/2)$ pairs of hypotheses from H, according to the probability computation $P(h)$ given above.

  For each pair (h_i, h_j), produce two offspring (i.e., children) by applying the crossover operator. Then, add all the offspring to H^{next}.

  Mutation

  Select $(r_{mu}.n)$ hypotheses of H_{next}, with uniform probability.

  For each selected hypothesis, invert one randomly chosen bit (i.e., 0 to 1, or 1 to 0) in the hypothesis's representation or 1 to 0)in the hypothesis's representation.

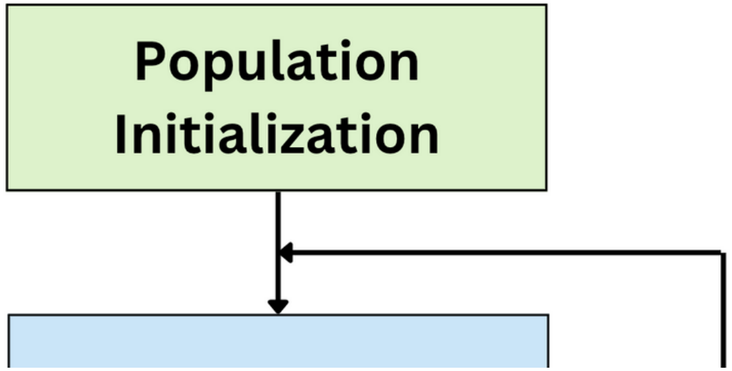
  Producing the next generation: $H \leftarrow H^{next}$

  Evaluate the new population. For each $h \in H$: compute Fitness(h)

return $argmax_{h \in H} Fitness(h)$;

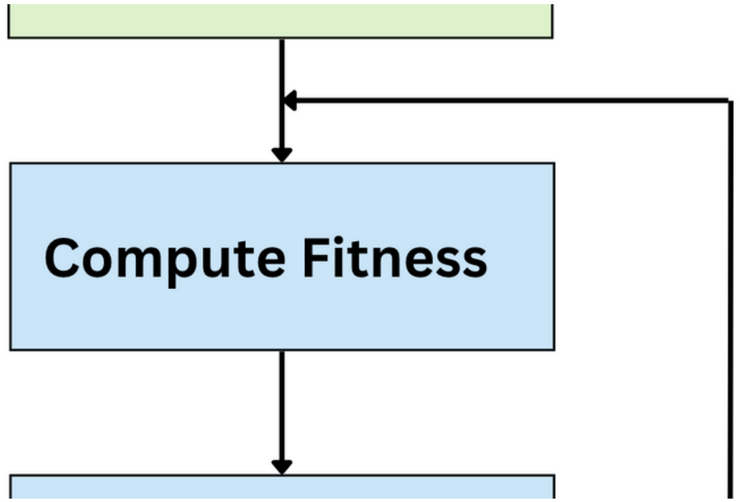
Khởi tạo quần thể (Population Initialization)

Tạo ra một tập hợp các cá thể ngẫu nhiên để đại diện cho các giải pháp có thể trong không gian tìm kiếm



Đánh giá cá thể (Compute Fitness)

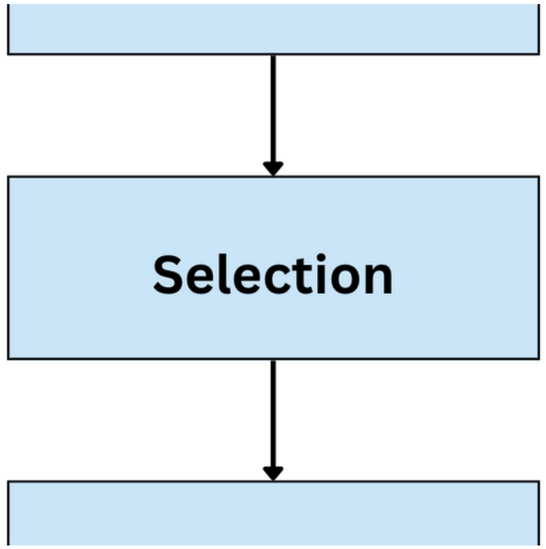
Sử dụng một hàm mục tiêu để đánh giá hiệu suất của mỗi cá thể trong quần thể.



UEH UNIVERSITY

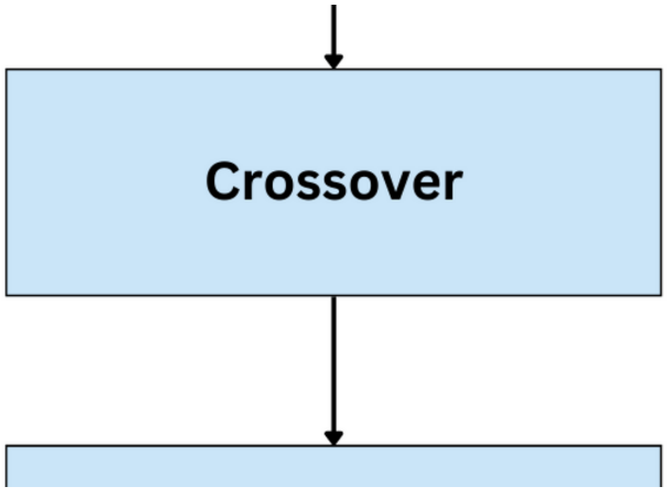
Lựa chọn (Selection)

Chọn ra một số lượng cá thể tốt nhất để tiếp tục phát triển trong thế hệ kế tiếp.



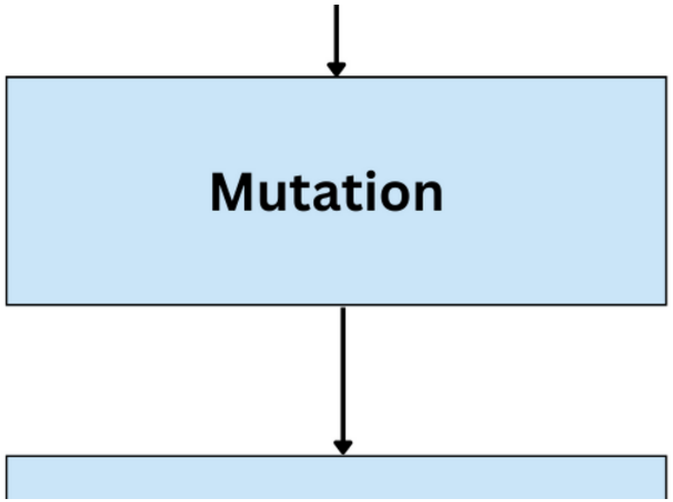
Lai ghép (Crossover)

Sử dụng các toán tử lai ghép để tạo ra các cá thể mới từ các cá thể đã chọn trong bước trước đó.



Đột biến (Mutation)

Áp dụng các toán tử đột biến để tạo ra sự đa dạng cho quần thể bằng cách thay đổi các gene của các cá thể.



UEH
UNIVERSITY

CÀI ĐẶT THUẬT TOÁN

UEH UNIVERSITY

Khởi tạo quần thể (Population Initialization)

```python
def generate_random_value():
    return random.randint(0, 1)
```

```python
def create_individual():
    return [generate_random_value() for _ in range(n)]
```

UEH UNIVERSITY

Đánh giá cá thể (Compute Fitness)

```python
def compute_fitness(chromosome, values, weights, max_weight):
    value = sum([chromosome[i] * values[i] for i in range(len(chromosome))])
    weight = sum([chromosome[i] * weights[i] for i in range(len(chromosome))])
    if weight > max_weight:
        return 0
    else:
        return value
```

UEH UNIVERSITY

Lựa chọn (Selection)

```
1  def selection(sorted_population):
2      index1 = random.randint(0, m-1)
3      while True:
4          index2 = random.randint(0, m-1)
5          if index2 != index1:
6              break
7      individual = sorted_population[index1]
8      if index1 < index2:
9          individual = sorted_population[index2]
10     return individual
```

UEH UNIVERSITY

Lai ghép (Crossover)

```python
def crossover(parent1, parent2):
    split_index = random.randint(1, len(parent1)-1)
    child1 = parent1[:split_index] + parent2[split_index:]
    child2 = parent2[:split_index] + parent1[split_index:]
    return child1, child2
```

UEH UNIVERSITY

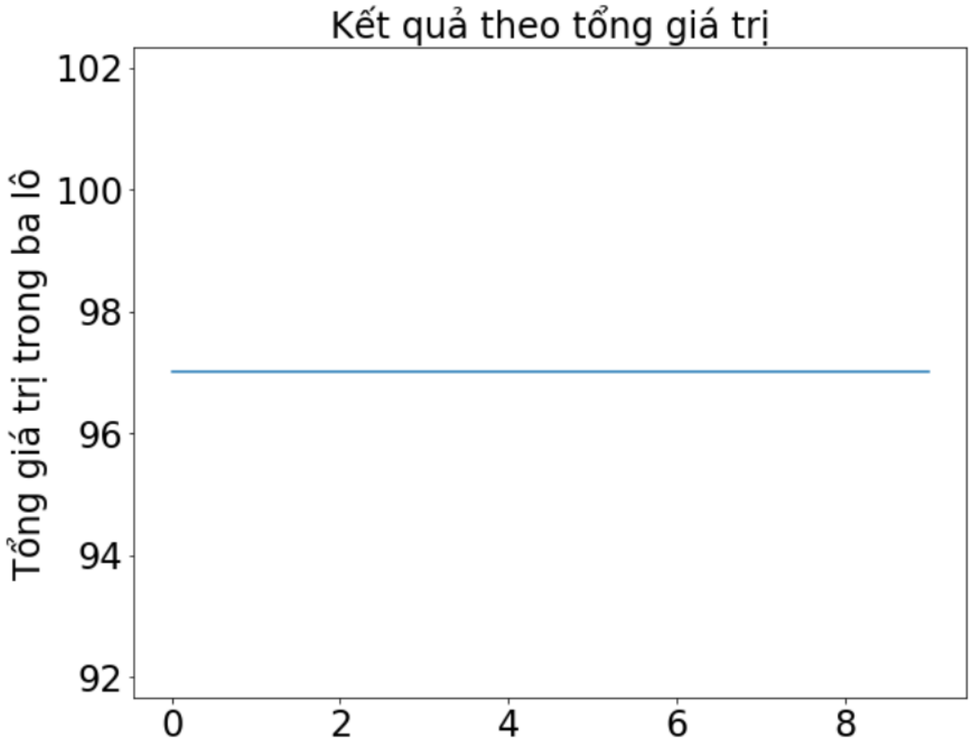
Đột biến (Mutation)

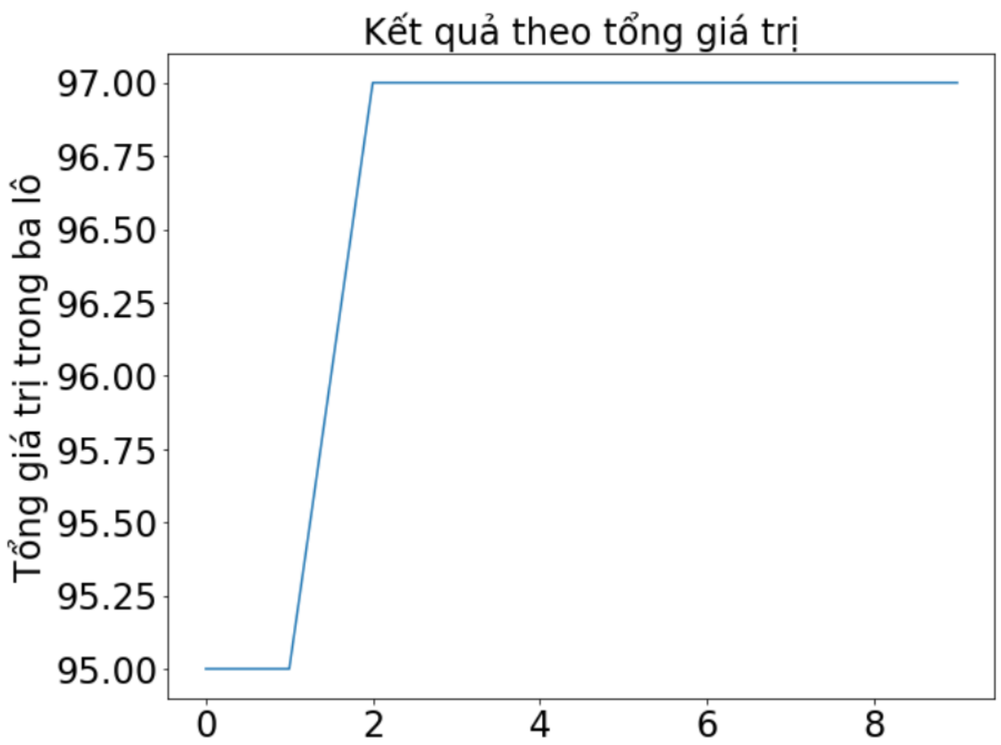
```
def mutate(individual, mutation_rate = 0.05):
    individual_new = individual.copy()
    if random.random() < mutation_rate:
        index = random.randint(0, n-1)
        individual_new[index] = generate_random_value()
    return individual_new
```

UEH
UNIVERSITY

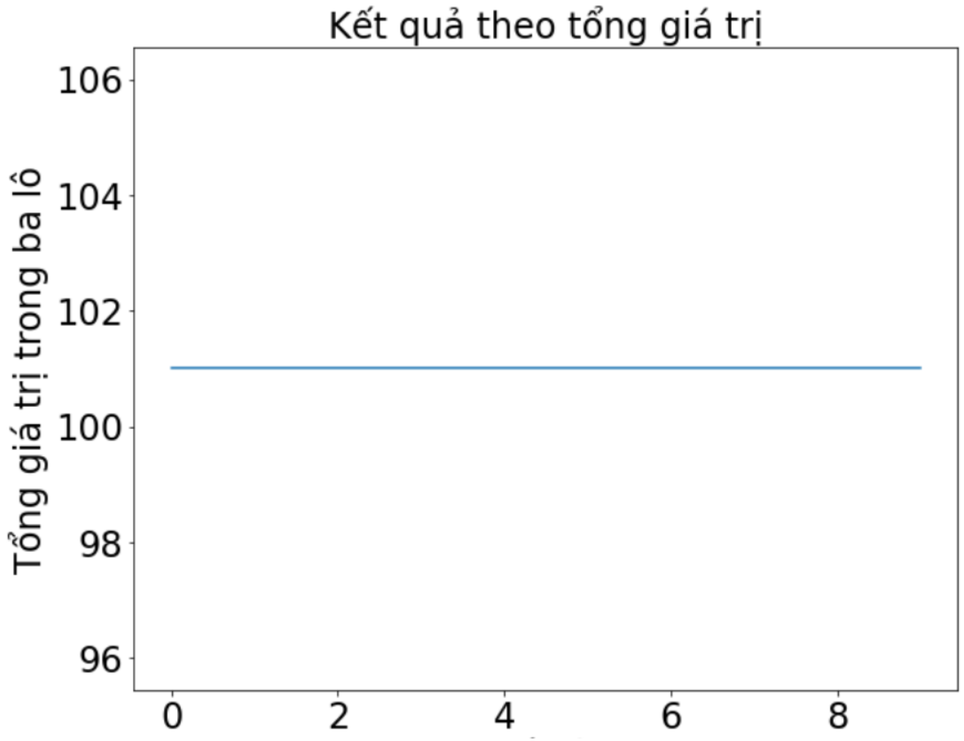
KẾT QUẢ THỰC NGHIỆM

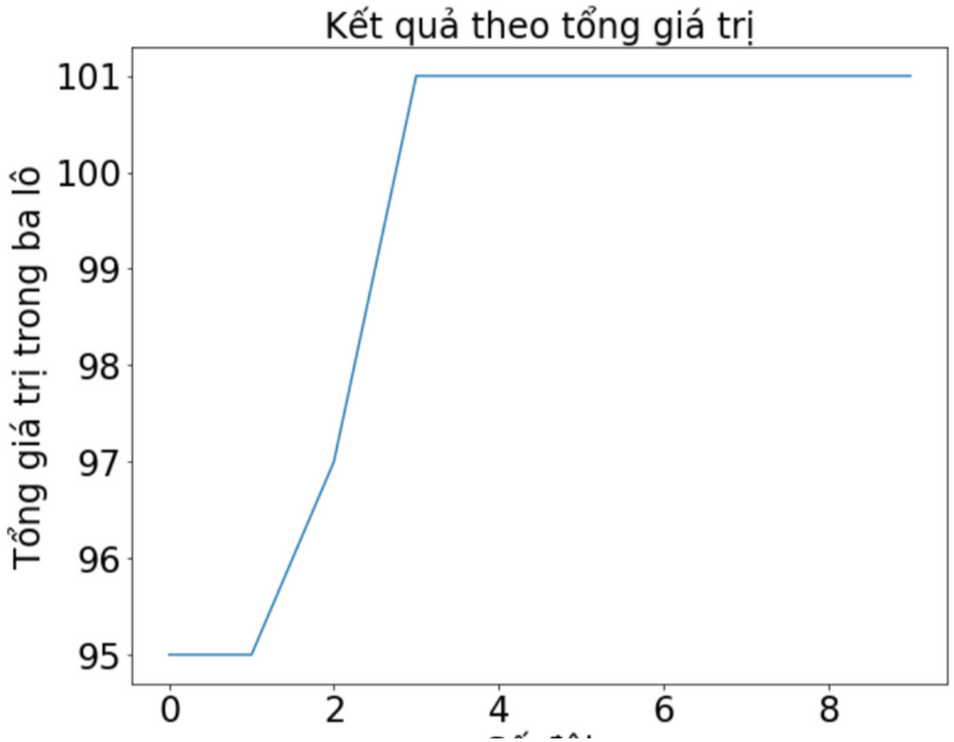
Các vật cho vào túi: [1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0] với khối lượng là 70 và giá tiền là **97**.





Các vật cho vào túi: [0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0] với khối lượng là 70 và giá tiền là **101**.

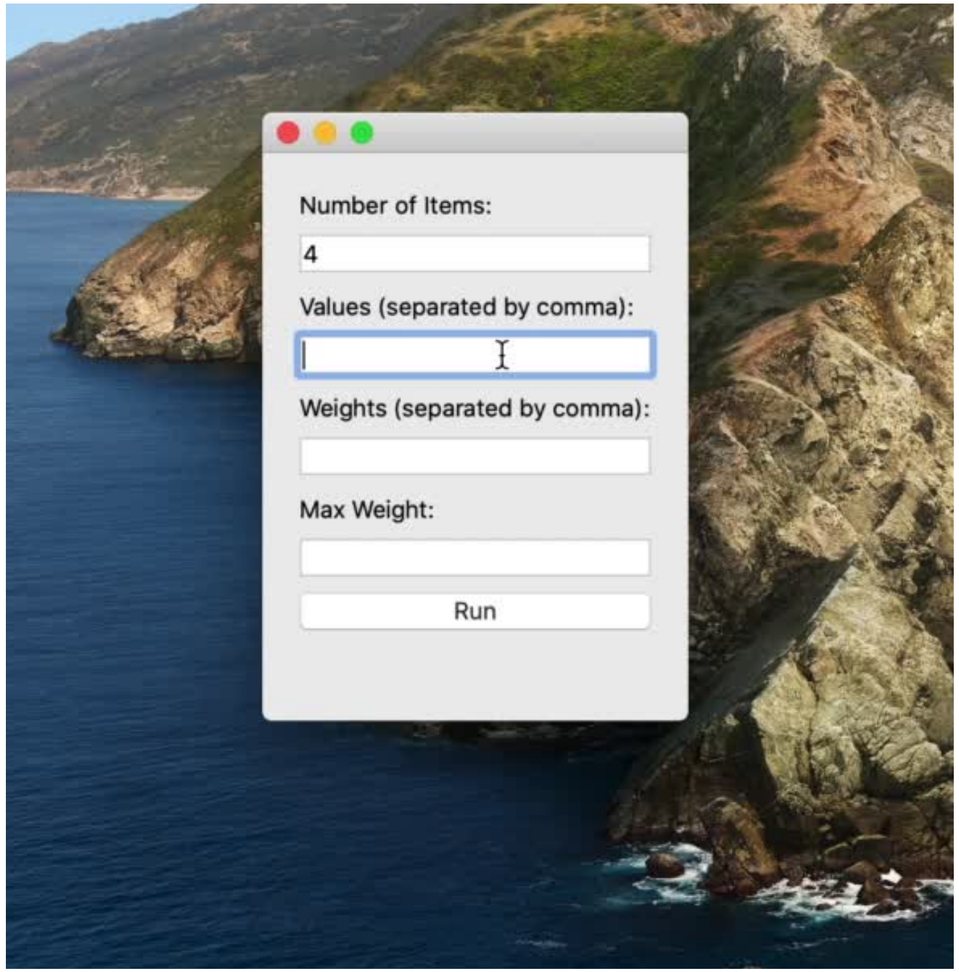




UEH
UNIVERSITY

GIAO DIỆN

Người dùng nhập vào số lượng các vật phẩm, giá trị và trọng lượng tương ứng và trọng lượng tối đa. Sau khi nhất nút **"Run"** thuật toán sẽ được thực thi và kết quả các vật phẩm được chọn, tổng giá trị và tổng số kí tương ứng sẽ được hiển thị lên màn hình.



UEH
UNIVERSITY

ỨNG DỤNG THỰC TIỄN

UEH UNIVERSITY

Tối ưu hóa kế hoạch sản xuất	Giải thuật di truyền được sử dụng để tối ưu hóa kế hoạch sản xuất trong các nhà máy. Quá trình tiến hóa được sử dụng để tìm ra kế hoạch sản xuất tối ưu dựa trên các yêu cầu về nguồn lực, thời gian và chi phí.
Tối ưu hóa tuyến đường vận chuyển	Giải thuật di truyền được sử dụng để tối ưu hóa tuyến đường vận chuyển trong các hệ thống giao thông. Quá trình tiến hóa được sử dụng để tìm ra tuyến đường vận chuyển tối ưu dựa trên các yêu cầu về thời gian và chi phí.