```
fitness_function = "(x ** 3) - 8 * (x ** 2) + 4"
```

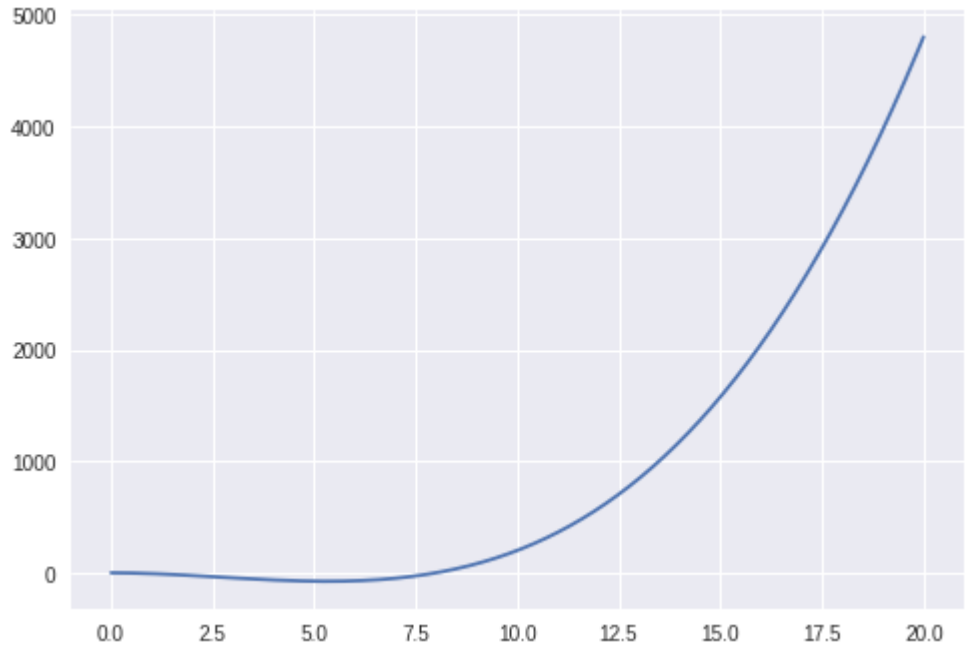
```
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('seaborn')
```

```
ip = input("Enter the range: ")
l, h = [int(val) for val in ip.split(' ')]
```

```
Enter the range: 0 20
```

```
arr = []
x = l
x_axis = []
for i in range((h - l) * 100):
  x += 0.01
  x_axis.append(x)
  arr.append(eval(fitness_function))
plt.plot(x_axis, arr)
```

```
[<matplotlib.lines.Line2D at 0x7f7c051322e8>]
```




```
def get_fitness(x):
  x = int(x, 2)
  return eval(fitness_function)
```

```
def initialize_population(n=6):
  return [bin(np.random.randint(l + 1, h + 1)).lstrip('0b') for i in range(n)]
initialize_population()
```

```
['1111', '1111', '1001', '101', '111', '110']
```

```
def mutation(gene):
```

```
  idx = np.random.randint(len(gene))
  return "".join([val if i != idx else '1' for i, val in enumerate(gene)])
```

```
def crossover(gene1, gene2):
  if len(gene1) > len(gene2):
    gene1, gene2 = gene2, gene1
  gene1 = '0' * (len(gene2) - len(gene1)) + gene1
  n = len(gene1)
  temp1, temp2 = gene1[: n // 2]  + gene2[n // 2:], gene1[n // 2:]  + gene2[:n // 2
  ans = []
  if l <= int(temp1, 2) <= h:
    ans.append(temp1)
  if l <= int(temp2, 2) <= h:
    ans.append(temp2)
  return ans
```

```
def get_next_generation(pop, n=3, mutation_rate=0.01):
  fitness = [get_fitness(val) for val in pop]
  pop_fitness = [{'info': val, 'fitness': fitness[i]} for i, val in enumerate(pop)]
  pop_fitness.sort(key=lambda val: val['fitness'])
  top_genes = pop_fitness[-n:]
  next_gen = [] # [val['info'] for val in top_genes]
  for i in range(n):
    for j in range(i + 1, n):
      t = crossover(pop[i], pop[j])
      if t:
        next_gen += t
      if np.random.rand() <= mutation_rate:
        next_gen[-1] = mutation(next_gen[-1])
  return next_gen, top_genes[-1]['fitness']
```

```
def main():
  population = initialize_population()
  print("Initial Population: ")
  print(population)
  fitness_for_generations = []
  num_iterations = 100
  for i in range(num_iterations):
    population, max_fitness = get_next_generation(population)
    fitness_for_generations.append(max_fitness)
  fitness_for_generations.sort()
  for i in range(num_iterations):
    print(f"Generation: {i}, Max Fitness: {fitness_for_generations[i]}")
  plt.title('Max Fitness over Generations')
  plt.xlabel('Generations')
  plt.ylabel('Best Fitness')
  plt.plot(fitness_for_generations)
  # return fitness_for_generations
```

```
main()
```

```
Initial Population:
['1000', '1001', '10011', '1', '110', '111']
Generation: 0, Max Fitness: 367
Generation: 1, Max Fitness: 367
Generation: 2, Max Fitness: 367
Generation: 3, Max Fitness: 367
Generation: 4, Max Fitness: 367
Generation: 5, Max Fitness: 367
Generation: 6, Max Fitness: 367
Generation: 7, Max Fitness: 367
Generation: 8, Max Fitness: 367
Generation: 9, Max Fitness: 367
Generation: 10, Max Fitness: 367
Generation: 11, Max Fitness: 367
Generation: 12, Max Fitness: 367
Generation: 13, Max Fitness: 367
Generation: 14, Max Fitness: 367
Generation: 15, Max Fitness: 367
Generation: 16, Max Fitness: 367
Generation: 17, Max Fitness: 367
Generation: 18, Max Fitness: 367
Generation: 19, Max Fitness: 367
Generation: 20, Max Fitness: 367
Generation: 21, Max Fitness: 367
Generation: 22, Max Fitness: 367
Generation: 23, Max Fitness: 367
Generation: 24, Max Fitness: 367
Generation: 25, Max Fitness: 367
Generation: 26, Max Fitness: 367
Generation: 27, Max Fitness: 367
Generation: 28, Max Fitness: 367
Generation: 29, Max Fitness: 580
Generation: 30, Max Fitness: 849
Generation: 31, Max Fitness: 849
Generation: 32, Max Fitness: 849
Generation: 33, Max Fitness: 849
Generation: 34, Max Fitness: 849
Generation: 35, Max Fitness: 849
Generation: 36, Max Fitness: 849
Generation: 37, Max Fitness: 849
Generation: 38, Max Fitness: 849
Generation: 39, Max Fitness: 849
Generation: 40, Max Fitness: 849
Generation: 41, Max Fitness: 849
Generation: 42, Max Fitness: 849
Generation: 43, Max Fitness: 849
Generation: 44, Max Fitness: 849
Generation: 45, Max Fitness: 849
Generation: 46, Max Fitness: 849
Generation: 47, Max Fitness: 849
Generation: 48, Max Fitness: 849
Generation: 49, Max Fitness: 849
Generation: 50, Max Fitness: 849
Generation: 51, Max Fitness: 849
Generation: 52, Max Fitness: 849
Generation: 53, Max Fitness: 849
Generation: 54, Max Fitness: 849
Generation: 55, Max Fitness: 849
Generation: 56, Max Fitness: 849
Generation: 57, Max Fitness: 849
Generation: 58, Max Fitness: 849
```

```
Generation: 59, Max Fitness: 1180
Generation: 60, Max Fitness: 1180
Generation: 61, Max Fitness: 1180
Generation: 62, Max Fitness: 1180
Generation: 63, Max Fitness: 1180
Generation: 64, Max Fitness: 1180
Generation: 65, Max Fitness: 1180
Generation: 66, Max Fitness: 1579
Generation: 67, Max Fitness: 1579
Generation: 68, Max Fitness: 1579
Generation: 69, Max Fitness: 1579
Generation: 70, Max Fitness: 1579
Generation: 71, Max Fitness: 1579
Generation: 72, Max Fitness: 1579
Generation: 73, Max Fitness: 2605
Generation: 74, Max Fitness: 2605
Generation: 75, Max Fitness: 2605
Generation: 76, Max Fitness: 2605
Generation: 77, Max Fitness: 2605
Generation: 78, Max Fitness: 2605
Generation: 79, Max Fitness: 2605
Generation: 80, Max Fitness: 2605
Generation: 81, Max Fitness: 2605
Generation: 82, Max Fitness: 2605
Generation: 83, Max Fitness: 2605
Generation: 84, Max Fitness: 2605
Generation: 85, Max Fitness: 2605
Generation: 86, Max Fitness: 2605
Generation: 87, Max Fitness: 2605
Generation: 88, Max Fitness: 2605
Generation: 89, Max Fitness: 2605
Generation: 90, Max Fitness: 2605
Generation: 91, Max Fitness: 2605
Generation: 92, Max Fitness: 2605
Generation: 93, Max Fitness: 2605
Generation: 94, Max Fitness: 2605
Generation: 95, Max Fitness: 2605
Generation: 96, Max Fitness: 2605
Generation: 97, Max Fitness: 2605
Generation: 98, Max Fitness: 2605
Generation: 99, Max Fitness: 3975
```