DATE:

Name: Prerna Sunil Jadhav
Sap ID: 60004220127
Batch: C22
Course: Advance Algorithm

EXP 2

AIM: The Hiring Problem

THEORY: The Hiring problem, also known as the secretary problem or the marriage problem is a classic problem in the analysis of algorithms and decision theory.
Imagine you need to hire a secretary from a pool of applicants. You interview them one by one in a random order. After each interview you must decide whether to hire the current candidate or move onto the next one. If you choose to hire a candidate the process ends. If you reject a candidate, you cannot go back to them later.
The goal is to maximize the probability of hiring the best candidate.
The problem demonstrate the balance between exploration and exploitation choosing the best option so far.
The optimal strategy known as the "37% rule" suggests that you should reject the

first 37% of candidates and hire the first candidate who is better than any of the previous ones.

The hiring problem has applications in various fields, including personnel selection, optimal stopping theory, and even in dating its a fascinating problem, because it illustrates the trade offs involved in decision making under uncertainty and limited information.

CONCLUSION: Hence, we studied and implemented hiring problem.

Name:	Prerna Sunil Jadhav
Sap Id:	60004220127
Class:	T. Y. B. Tech (Computer Engineering)
Course:	Advance Algorithm Laboratory
Course Code:	DJ19CEL602
Experiment No.:	02

AIM: Hiring Problem

CODE:

Ascending:

```
import random
candidate = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
interview = []
hire = []

for i in range(0, 10):
  x = random.choice(candidate)
  candidate.remove(x)
  interview.append(x)

print(interview)
for i, num in enumerate(interview, 1):
  largest_num = max(interview[:i])
  print(f"Hired: {largest_num}, till {i} interviews")
  hire.append(largest_num)

print(hire)
print("The number of candidates hired is:", len(set(hire)))
cost = len(set(hire)) - 1
print("Thus firing cost will be:", cost)
```

OUTPUT:

```
[5, 8, 4, 0, 7, 9, 2, 3, 1, 6]
Hired: 5, till 1 interviews
Hired: 8, till 2 interviews
Hired: 8, till 3 interviews
Hired: 8, till 4 interviews
Hired: 8, till 5 interviews
Hired: 9, till 6 interviews
Hired: 9, till 7 interviews
Hired: 9, till 8 interviews
Hired: 9, till 9 interviews
Hired: 9, till 10 interviews
[5, 8, 8, 8, 8, 9, 9, 9, 9, 9]
The number of candidates hired is: 3
Thus firing cost will be: 2
```

Randomized:

```
import random

candidates = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
interviewed_candidates = []
hired_candidates = []

# Randomly select and interview candidates
for i in range(len(candidates)):
    selected_candidate = random.choice(candidates)
    interviewed_candidates.append(selected_candidate)
    candidates.remove(selected_candidate)

# Hire the best candidate so far
max=-1
for i in range(len(interviewed_candidates)):
    if interviewed_candidates[i] > max:
        max=interviewed_candidates[i]
        hired_candidates.append(interviewed_candidates[i])

# Calculate firing cost
firing_cost = len(hired_candidates) - 1

print("Interviewed candidates:", interviewed_candidates)
print("Hired candidates:", hired_candidates)
print("Number of candidates hired:", len(hired_candidates))
print("Firing cost:", firing_cost)
```

OUTPUT:

```
Interviewed candidates: [6, 8, 1, 2, 4, 5, 9, 7, 0, 3]
Hired candidates: [6, 8, 9]
Number of candidates hired: 3
Firing cost: 2
```

CONCLUSION: Hence we implemented the Hiring Problem.