Institute of Computer Technology

B. Tech Computer Science and Engineering

Sub: Algorithm Analysis and Design

Practical 2

(1) MPSoft Technologies Pvt. Ltd. is a fast growing IT industry and wants to implement a function to calculate the monthly income generated from all projects from their N no of clients like C1,C2,C3,C4….CN. The team wants to compare the time/steps required to execute this function on various inputs and analyse the complexity of each combination. Also draw a comparative chart. In each of the following functions N will be passed by user.

Design the algorithm for the same and implement using the programming language of your choice. Make comparative analysis for various use cases & input size.

1. **To calculate the sum of 1 to N number using loop.**
2. **To calculate the sum of 1 to N number using the equation.**
3. **To calculate sum of 1 to N numbers using recursion**

Code:

App.py

```
from flask import Flask, render_template, request
import matplotlib.pyplot as plt
import os

import matplotlib
matplotlib.use('Agg')
```

```
app = Flask(__name__)

count_loop = 0
count_eqn = 0
count_rec = 0

def sum_using_loop(n):
    global count_loop
    total = 0
    for i in range(1, n + 1):
        total += i
        count_loop += 1
    return total

def sum_using_equation(n):
    global count_eqn
    count_eqn += 1
    return n * (n + 1) // 2

def sum_using_recursion(n):
    global count_rec
    count_rec += 1
    if n == 1:
        count_rec += 1
        return 1
    else:
        count_rec += 1
        return n + sum_using_recursion(n - 1)

def compare_methods(inputs):
```

```
    global count_loop, count_eqn, count_rec
    results = []
    for n in inputs:
        count_loop = 0
        count_eqn = 0
        count_rec = 0

        sum_using_loop(n)
        loop_count = count_loop

        sum_using_equation(n)
        eqn_count = count_eqn

        try:
            sum_using_recursion(n)
            rec_count = count_rec
        except RecursionError:
            rec_count = 'Infinity'

        results.append((n, loop_count, eqn_count, rec_count))
    return results

def plot_results(results):
    n_values = [result[0] for result in results]
    loop_counts = [result[1] for result in results]
    eqn_counts = [result[2] for result in results]
    rec_counts = [result[3] if result[3] != 'Infinity' else max(loop_counts) * 1.1 for result in results]

    plt.figure(figsize=(7, 4))
    plt.plot(n_values, loop_counts, label='Loop Count', color='lightblue', marker='o')
    plt.plot(n_values, eqn_counts, label='Equation Count', color='lightgreen', marker='o')
```

```
    plt.plot(n_values, rec_counts, label='Recursion Count', color='lightcoral', marker='o')
    plt.xlabel('N')
    plt.ylabel('Count of Steps')
    plt.title('Comparison of Sum Calculation Methods by Step Count')
    plt.legend()
    plt.grid(True)
    plt.tight_layout()
    plot_path = os.path.join('static', 'plot.png')
    plt.savefig(plot_path)
    plt.close()


@app.route("/", methods=["GET", "POST"])
def index():
    if request.method == "POST":
        inputs = request.form.get("inputs")
        inputs = list(map(int, inputs.split()))
        results = compare_methods(inputs)
        plot_results(results)
        complexities = {
            "Loop Count": "O(N)",
            "Equation Count": "O(1)",
            "Recursion Count": "O(N)"
        }
        return render_template("index.html", results=results, complexities=complexities)
    return render_template("index.html")


if __name__ == "__main__":
    app.run(debug=True)
```

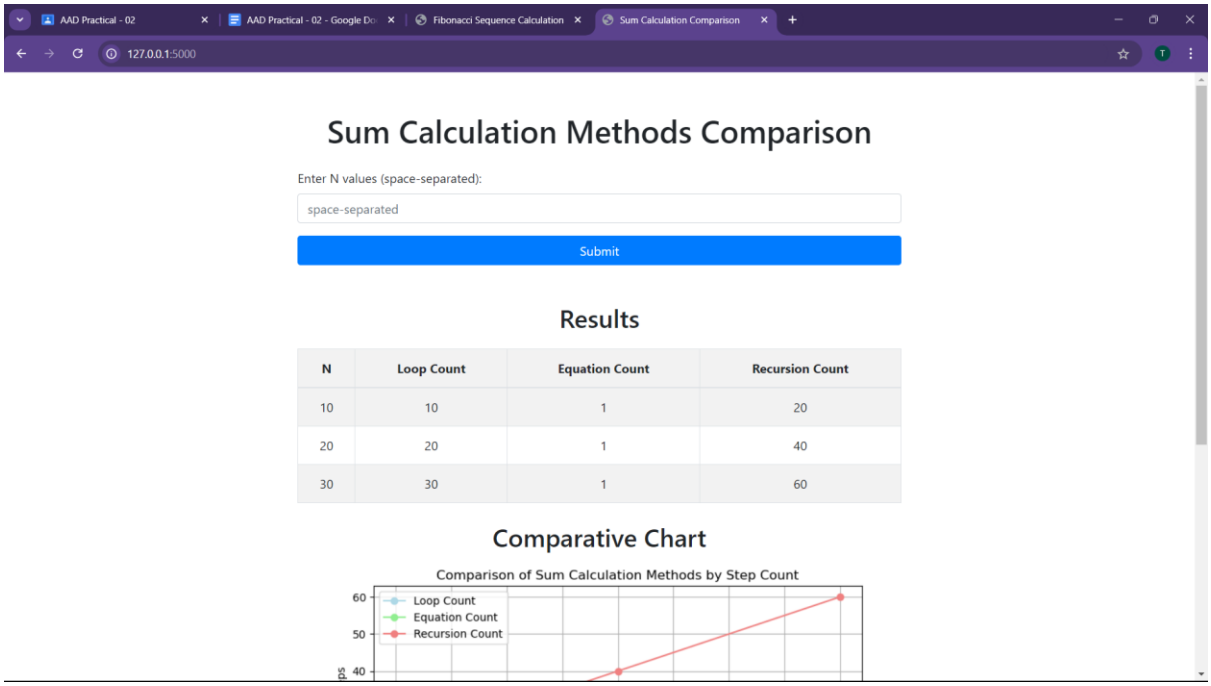
Index.html

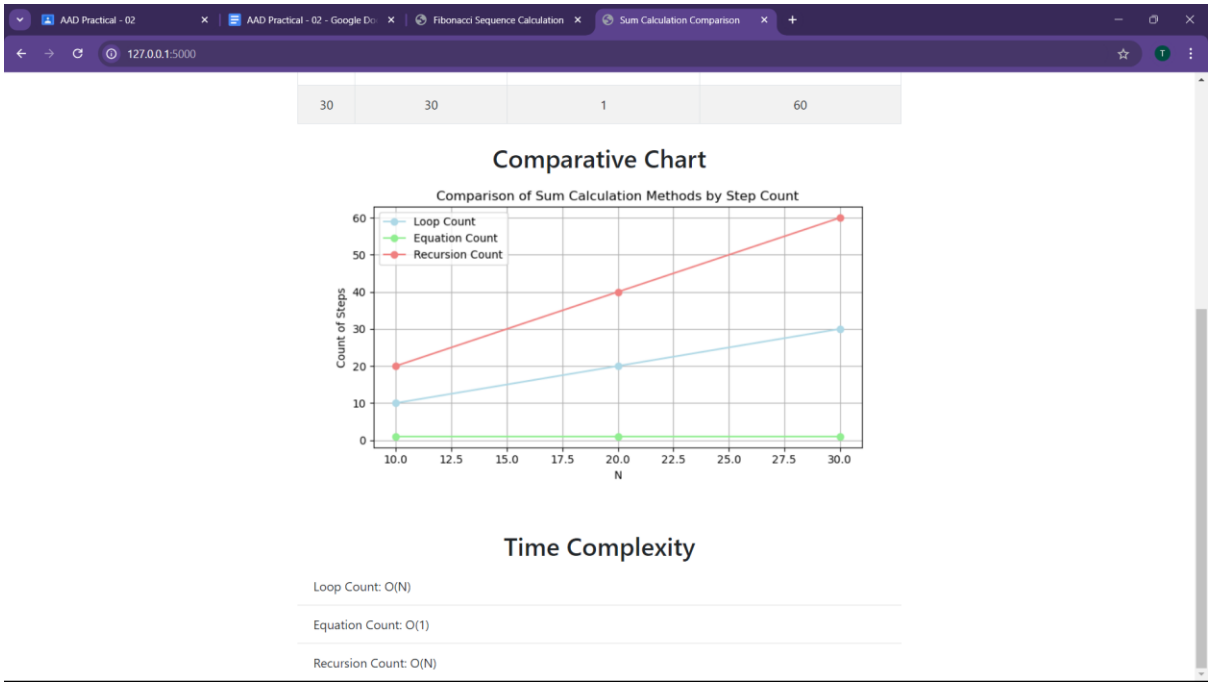
```
<!DOCTYPE html>
```

```
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Sum Calculation Comparison</title>
    <!-- Bootstrap CSS -->
    <link href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"
rel="stylesheet">
    <style>
        .container {
            max-width: 800px;
            margin-top: 50px;
        }
        .chart-container {
            text-align: center;
            margin-top: 20px;
        }
        table {
            margin-top: 20px;
            border-collapse: collapse;
        }
        th, td {
            padding: 8px 12px;
            text-align: center;
        }
        .table th {
            background-color: #f2f2f2;
        }
    </style>
</head>
<body>
```

```
  <div class="container">
    <h1 class="text-center">Sum Calculation Methods Comparison</h1>
    <form method="post" class="mt-4">
      <div class="form-group">
        <label for="inputs">Enter N values (space-separated):</label>
        <input type="text" class="form-control" id="inputs" name="inputs" placeholder="space-separated" required>
      </div>
      <button type="submit" class="btn btn-primary btn-block">Submit</button>
    </form>

    {% if results %}
    <h2 class="text-center mt-5">Results</h2>
    <table class="table table-bordered table-striped">
      <thead>
        <tr>
          <th>N</th>
          <th>Loop Count</th>
          <th>Equation Count</th>
          <th>Recursion Count</th>
        </tr>
      </thead>
      <tbody>
        {% for result in results %}
        <tr>
          <td>{{ result[0] }}</td>
          <td>{{ result[1] }}</td>
          <td>{{ result[2] }}</td>
          <td>{{ result[3] }}</td>
        </tr>
        {% endfor %}
```

```
        </tbody>
    </table>

    <div class="chart-container">
        <h2 class="text-center mt-4">Comparative Chart</h2>
        <img src="{{ url_for('static', filename='plot.png') }}" alt="Comparison Chart" class="img-fluid">
    </div>

    <h2 class="text-center mt-5">Time Complexity</h2>
    <ul class="list-group list-group-flush">
        <li class="list-group-item">Loop Count: {{ complexities['Loop Count'] }}</li>
        <li class="list-group-item">Equation Count: {{ complexities['Equation Count'] }}</li>
        <li class="list-group-item">Recursion Count: {{ complexities['Recursion Count'] }}</li>
    </ul>
    {% endif %}
  </div>

  <!-- Bootstrap JS and dependencies -->
  <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
  <script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.4/dist/umd/popper.min.js"></script>
  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</body>
</html>
```





(2) Suppose a newly-born pair of rabbits, one male, one female, are put in a field. Rabbits are able to mate at the age of one month so that at the end of its second month a female can produce another pair of rabbits. Suppose that our rabbits never die and that the female always produces one new pair (one male, one female) every month from the second month on. How many pairs will there be in one year? Apply appropriate algorithm/method to find out

the above problem and also solve them using iteration and recursive method. Compare the performance of two methods by counting the number of steps executed on various inputs. Also draw a comparative chart.

Design the algorithm for the same and implement using the programming language of your choice. Make comparative analysis for various use cases & input size.

Code:

App.py

```
from flask import Flask, render_template, request
import matplotlib.pyplot as plt
import os
import numpy as np
from scipy.interpolate import make_interp_spline
import matplotlib
matplotlib.use('Agg')


app = Flask(__name__)


def fibonacci_iterative(n):
    a, b = 0, 1
    steps = 0
    for _ in range(n):
        a, b = b, a + b
        steps += 1
    return b, steps


def fibonacci_recursive(n, steps=0):
    if n <= 1:
        return n, steps + 1
    else:
        fib1, steps1 = fibonacci_recursive(n - 1, steps + 1)
```

```
    fib2, steps2 = fibonacci_recursive(n - 2, steps1)
    return fib1 + fib2, steps2

def compare_methods(max_month):
  results = []
  for month in range(max_month + 1):
    iter_value, iter_steps = fibonacci_iterative(month)
    rec_value, rec_steps = fibonacci_recursive(month)
    results.append((month, iter_value, iter_steps, rec_value, rec_steps))
  return results

def plot_results(results):
  months = np.array([result[0] for result in results])
  iter_steps = np.array([result[2] for result in results])
  rec_steps = np.array([result[4] for result in results])

  xnew = np.linspace(months.min(), months.max(), 300)

  spl_iter = make_interp_spline(months, iter_steps, k=3)
  spl_rec = make_interp_spline(months, rec_steps, k=3)

  iter_smooth = spl_iter(xnew)
  rec_smooth = spl_rec(xnew)

  plt.figure(figsize=(10, 5))

  plt.plot(xnew, iter_smooth, label='Iterative Steps', linestyle='-', color='royalblue', linewidth=2)
  plt.plot(xnew, rec_smooth, label='Recursive Steps', linestyle='-', color='tomato', linewidth=2)

  plt.xlabel('Month')
  plt.ylabel('Steps')
```

```
    plt.title('Comparison of Iterative and Recursive Methods')
    plt.legend()
    plt.grid(True)
    plt.tight_layout()

    plot_path = os.path.join('static', 'plot.png')
    plt.savefig(plot_path)
    plt.close()
@app.route("/", methods=["GET", "POST"])
def index():
    error = None
    results = None
    if request.method == "POST":
        max_month_str = request.form.get("max_month")
        if max_month_str and max_month_str.isdigit():
            max_month = int(max_month_str)
            results = compare_methods(max_month)
            plot_results(results)
        else:
            error = "Please enter a valid number of months."
    return render_template("index2.html", results=results, error=error)

if __name__ == "__main__":
    app.run(debug=True)
```

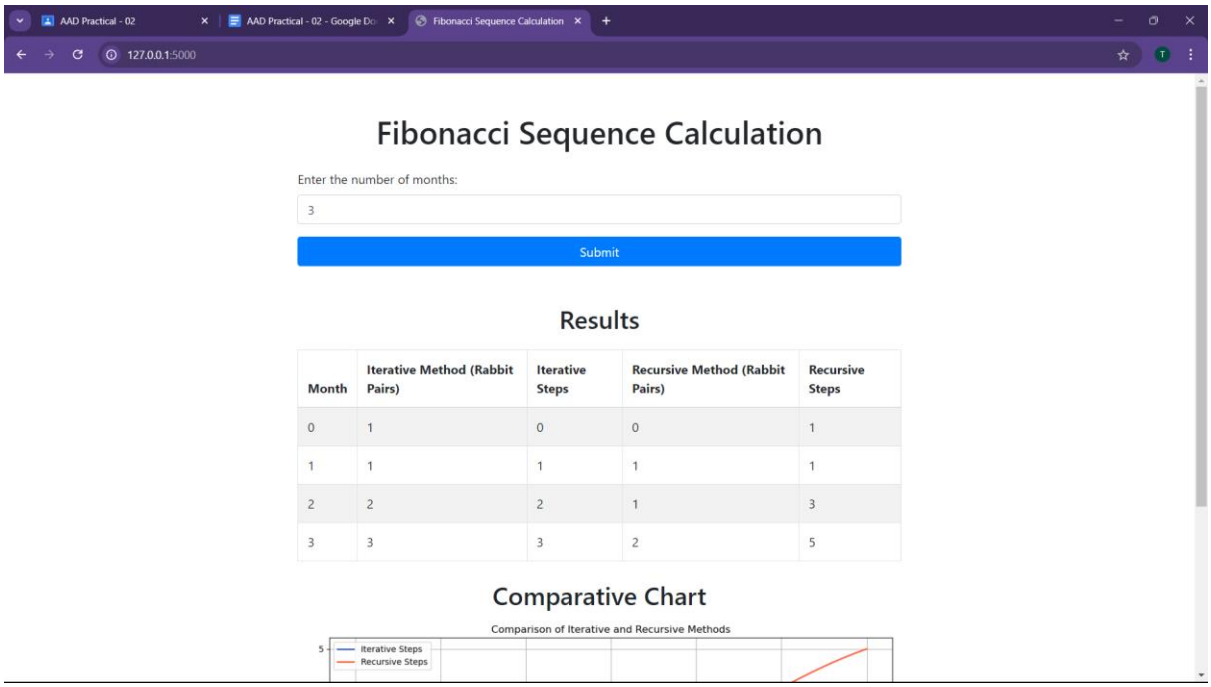
Index2.html

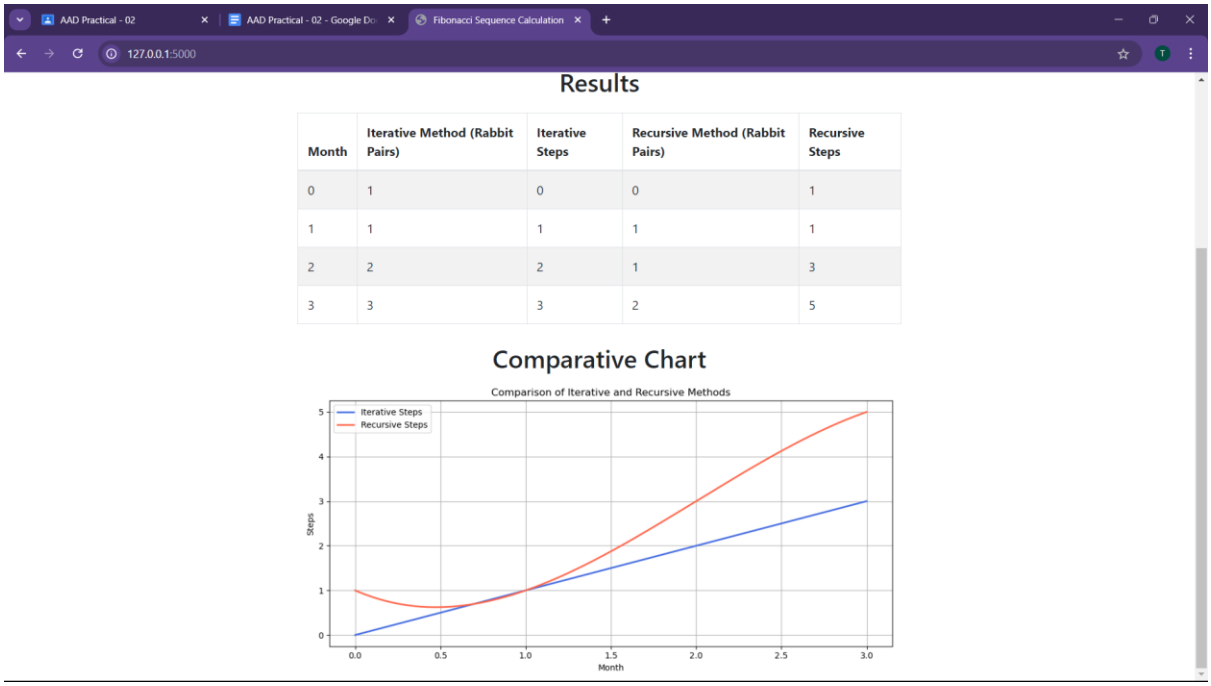
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <title>Fibonacci Sequence Calculation</title>
    <link href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" rel="stylesheet">
    <style>
        .container {
            max-width: 800px;
            margin-top: 50px;
        }
        .chart-container {
            text-align: center;
            margin-top: 20px;
        }
        .table {
            margin-top: 20px;
        }
    </style>
</head>
<body>
    <div class="container">
        <h1 class="text-center">Fibonacci Sequence Calculation</h1>
        <form method="post" class="mt-4">
            <div class="form-group">
                <label for="max_month">Enter the number of months:</label>
                <input type="text" class="form-control" id="max_month" name="max_month" placeholder="space-separated" required>
            </div>
            <button type="submit" class="btn btn-primary btn-block">Submit</button>
        </form>

        {% if error %}
        <div class="alert alert-danger mt-4">{{ error }}</div>
        {% endif %}
```

```
{% if results %}
<h2 class="text-center mt-5">Results</h2>
<table class="table table-bordered table-striped">
  <thead>
    <tr>
      <th>Month</th>
      <th>Iterative Method (Rabbit Pairs)</th>
      <th>Iterative Steps</th>
      <th>Recursive Method (Rabbit Pairs)</th>
      <th>Recursive Steps</th>
    </tr>
  </thead>
  <tbody>
    {% for result in results %}
    <tr>
      <td>{{ result[0] }}</td>
      <td>{{ result[1] }}</td>
      <td>{{ result[2] }}</td>
      <td>{{ result[3] }}</td>
      <td>{{ result[4] }}</td>
    </tr>
    {% endfor %}
  </tbody>
</table>

<div class="chart-container">
  <h2 class="text-center mt-4">Comparative Chart</h2>
  <img src="{{ url_for('static', filename='plot.png') }}" alt="Comparison Chart" class="img-fluid">
</div>
{% endif %}
```

```
    </div>


    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
    <script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.4/dist/umd/popper.min.js"></script>
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</body>
</html>
```




AAD Practical - 02
AAD Practical - 02 - Google Do
Fibonacci Sequence Calculation
127.0.0.1:5000
Results
Month
Iterative Method (Rabbit Pairs)
Iterative Steps
Recursive Method (Rabbit Pairs)
Recursive Steps
0 1 0 0 1
1 1 1 1 1
2 2 2 1 3
3 3 3 2 5
Comparative Chart
Comparison of Iterative and Recursive Methods
Iterative Steps
Recursive Steps
Steps
Month
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0
1
2
3
4
5