Got it!
You want **only theory**, **applications**, **conceptual questions** with **viva Q&A**, and **examples** for each of these 7 programs — for **practical exam preparation**.
I'll give you clean, short, and ready-to-revise notes.

Lp2 pract

---

1. Depth First Search (DFS) & Breadth First Search (BFS)

Theory:

- **DFS**: Explores as far as possible along each branch before backtracking. (Uses **stack** or **recursion**).
- **BFS**: Explores all neighbors at the present depth before moving deeper. (Uses **queue**).

Applications:

- Finding connected components.
- Solving puzzles like mazes.
- Web crawling.

Example:

```
sql

`Graph:  1 — 2
         |
         3
DFS (start=1): 1 → 2 → 3
BFS (start=1): 1 → 2 → 3
`
```

Viva Q&A:

- Q: What data structure is used in DFS?
  A: Stack or recursion.

- Q: In BFS?
  A: Queue.
- Q: Can DFS get stuck in infinite loop?
  A: Yes, without visited marking.

---

2. A* (A star) Algorithm

Theory:

- **A*** is a best-first search algorithm.
- It uses **f(n) = g(n) + h(n)** where:
  - **g(n)** = cost from start to node n
  - **h(n)** = estimated cost to goal (heuristic)

Applications:

- Game AI (finding shortest paths).
- GPS Navigation Systems.

Example: Finding shortest path in a grid (start to goal).

Viva Q&A:

- Q: What makes A* optimal?
  A: If heuristic is admissible (never overestimates).
- Q: Is A* faster than Dijkstra?
  A: Yes, if heuristic is good.

---

3. Greedy Algorithm (Choose 1)

Let's pick **Prim's MST Algorithm**.

Theory:

- **Prim's** finds **Minimum Spanning Tree (MST)** starting from any vertex.
- Greedily picks smallest edge connecting to tree.

Applications:

- Network design (roads, cables).
- Cluster analysis.

Example:

```
mathematica

`Graph edges: (A-B,2), (B-C,3), (A-C,1)
Prim's MST: Pick A-C (1), then B (2)
`
```

Viva Q&A:

- Q: Difference between Prim's and Kruskal's?
  A: Prim's grows a tree; Kruskal's grows forests and merges.
- Q: Does Prim's work on disconnected graphs?
  A: No, only connected graphs.

---

4. N-Queens Problem (Branch and Bound + Backtracking)

Theory:

- Place N queens on N×N board such that no two queens attack.
- **Backtracking**: Try options, backtrack on conflict.
- **Branch and Bound**: Prune unpromising paths early.

Applications:

- Puzzle solving.
- Optimization problems.

Example:
4 Queens Solution:

```
css

`. Q . .
. . . Q
Q . . .
```

```
. . Q .
```

Viva Q&A:

- Q: What is backtracking?
  A: Trying choices and undoing if wrong.
- Q: What is branch and bound?
  A: Prune paths that can't lead to solution.

---

5. Graph Coloring Problem (Branch and Bound + Backtracking)

Theory:

- Assign colors to graph nodes such that adjacent nodes have different colors.
- **Backtracking**: Try colors recursively.
- **Branch and Bound**: Cut if too many colors used.

Applications:

- Scheduling exams.
- Register allocation in compilers.

Example: 3 colors to color a triangle graph.

Viva Q&A:

- Q: Minimum colors needed for a cycle graph?
  A: 2 if even cycle, 3 if odd cycle.
- Q: What is chromatic number?
  A: Minimum number of colors needed.

---

6. Elementary Chatbot

Theory:

- A simple rule-based chatbot responds to basic inputs with predefined answers.

- No ML, just pattern matching.

Applications:

- Customer support bots.
- FAQ bots.

Example:

```
vbnet

`User: "Hi"
Bot: "Hello! How can I help you?"
`
```

Viva Q&A:

- Q: What is difference between rule-based and AI chatbot?
  A: Rule-based uses if-else; AI chatbots learn patterns.
- Q: Which libraries can be used for chatbots?
  A: NLTK, Rasa, etc.

---

7. Expert System (Hospitals or any)

Theory:

- **Expert system** mimics human expert decision making.
- Uses **rules** (if-then) and **inference engine**.

Applications:

- Medical diagnosis.
- Troubleshooting machines.

Example:

```
nginx

`IF fever AND cough THEN possibility of Flu
`
```

Viva Q&A:

- Q: What are components of expert system?
  A: Knowledge base + Inference engine + User interface.
- Q: Example of medical expert system?
  A: MYCIN (early system for blood infections).

✅ Summary

Sr. No.	Program	Concept	Application	Key Example
1	DFS, BFS	Graph Search	Web Crawling	Visit all nodes
2	A* Search	Informed Search	Games, Maps	f(n) = g(n) + h(n)
3	Prim's Algorithm	MST	Network Design	Pick min edges
4	N-Queens	Backtracking	Puzzle Solving	Safe Queen placement
5	Graph Coloring	CSP	Scheduling	3-color a graph
6	Chatbot	Rule-based Interaction	Customer service	Hello-Hi
7	Expert System	Rule-based Decision	Medical field	Diagnose disease

---

Would you also like me to prepare a **few 2–3 marks theory questions** and **MCQs** based on these for even quicker revision? 🎯
(Will take just a few more minutes!)

Alright!
You want a clean **theory-only** version — no steps — covering **Amazon EC2, Google App Engine, Salesforce**, along with **Cloud Computing basics** like virtualization, and short descriptions of **GCP**, **AWS**, and **Salesforce Developer** interfaces.
I'll keep everything organized, exam-ready, and crisp for you. 📚✨

📚 CLOUD COMPUTING PRACTICAL THEORY NOTES

🌐 Basic Topics of Cloud Computing (CC)

1. Virtualization

- **Virtualization** allows creating virtual (not actual) versions of hardware like servers, storage, and networks.
- **Hypervisor** (like VMware, KVM) enables multiple virtual machines (VMs) on a single physical machine.
- **Types**:
  - Server Virtualization
  - Storage Virtualization
  - Network Virtualization

Why important?

- Better resource utilization
- Cost savings
- Easy scalability

---

2. Cloud Service Models

- **IaaS (Infrastructure as a Service)**: Raw resources like servers, storage (e.g., AWS EC2, Google Compute Engine).
- **PaaS (Platform as a Service)**: Ready-to-use platforms for app development (e.g., Google App Engine, Heroku).
- **SaaS (Software as a Service)**: Software delivered over the internet (e.g., Gmail, Salesforce CRM).

---

3. Cloud Deployment Models

- **Public Cloud**: Available to everyone (AWS, GCP).
- **Private Cloud**: Used by one organization.
- **Hybrid Cloud**: Combination of public + private.

☁️ Case Study 1: Amazon EC2 (Elastic Compute Cloud)

What is EC2?

- Amazon EC2 provides scalable virtual servers ("instances") on the AWS cloud.
- You can rent VMs, configure OS, storage, and networking.

Features:

- Launch, terminate, and manage server instances.
- Auto-scaling and Load Balancing.
- Choice of OS (Linux, Windows).
- Pay-as-you-go pricing.

Applications:

- Hosting websites and applications.
- Running backend servers for mobile apps.
- Big data processing (Hadoop clusters).

Example Use:

- A startup uses EC2 to launch a web server for its website without investing in hardware.

🌐 Case Study 2: Google App Engine (GAE)

What is Google App Engine?

- A **Platform as a Service (PaaS)** by Google Cloud.
- Allows developers to deploy web apps and APIs without managing servers.

Features:

- Supports multiple programming languages (Python, Java, Go, PHP).
- Automatic scaling based on traffic.
- Built-in security and load balancing.

Applications:

- Hosting web apps, backend APIs.
- Building scalable mobile app backends.
- Prototyping apps quickly.

Installation & Configuration (Theory Only):

- Requires installing Google Cloud SDK.
- Projects and applications are managed via Google Cloud Console.

🛠️ Case Study 3: Creating an Application in Salesforce Using Apex

What is Salesforce.com?

- Salesforce is a **cloud-based CRM (Customer Relationship Management)** platform.
- Allows businesses to manage sales, service, marketing, and more.

What is Apex?

- Apex is Salesforce's **proprietary programming language**.
- It is similar to Java and used for writing custom business logic.

Features of Apex:

- Triggers (before/after database events).
- Classes and Objects.
- SOQL (Salesforce Object Query Language) for database operations.

Applications:

- Automating business processes.
- Building complex workflows and custom apps.

📦 Mini Project: Custom Application in Salesforce Cloud

Objective:

- Design and develop a mini project using Salesforce Developer Console or Platform.

Example Ideas:

- **Student Management System** (track students, courses, marks).
- **Inventory Management** (track products, stock, orders).

Features in Project:

- Custom objects (Tables).
- Custom fields (Attributes).
- Triggers (automated actions).
- Visualforce Pages (optional UI).

💻 Description of Interfaces

1. AWS Console (Amazon EC2)

- **Interface**: Web-based dashboard.
- **Features**:
  - Manage instances (start, stop, terminate).
  - Monitor instance health (CloudWatch).
  - Create and assign Security Groups (firewall).
  - Elastic IP management.
- **Main Tools**: EC2 Dashboard, IAM (Identity Access Management), S3 storage, CloudFormation templates.

---

2. GCP Console (Google App Engine)

- **Interface**: Google Cloud Console (GUI).
- **Features**:
  - Deploy apps with a few clicks.
  - View app logs (Stackdriver).
  - Traffic splitting between app versions.
  - Set up auto-scaling, firewall rules.
- **Main Tools**: App Engine Dashboard, Cloud Storage, Firestore, GCP Monitoring.

---

3. Salesforce Developer Console

- **Interface**: Built-in web IDE within Salesforce Org.
- **Features**:
  - Write Apex classes and triggers.
  - SOQL Query editor to fetch records.
  - Debug logs viewer.
  - Test execution and result viewing.
- **Main Tools**: Setup Menu → Developer Console → Apex Classes, Triggers, Visualforce Pages.

✅ Summary Table

Topic	What it is	Applications	Interface
Virtualization	Virtual servers	Efficient cloud computing	Hypervisor
AWS EC2	Rent virtual machines	Hosting apps, websites	AWS EC2 Dashboard
Google App Engine	Serverless platform	App backend, websites	GCP Console
Salesforce + Apex	CRM + Custom Logic	Automate sales, service	Salesforce Developer Console

---

📝 Viva and Conceptual Questions

- Q: What is virtualization and why is it important for cloud computing?
  A: Virtualization allows better use of physical hardware by creating virtual servers, making cloud services scalable and cost-effective.
- Q: What is the difference between IaaS and PaaS?
  A: IaaS provides raw infrastructure (e.g., EC2), PaaS provides a ready-to-de