Explain the structure of a CloudFormation template. **Follow-up:** What are the mandatory sections in a CloudFormation template?

CloudFormation Template Structure

A CloudFormation Template is usually written in **YAML** (or JSON), and typically includes these top-level sections:

1. **AWSTemplateFormatVersion** – Specifies the template version (optional, but recommended).
2. **Description** – Short info about what the template does.
3. **Metadata** – Additional data about the template (not processed by CloudFormation).
4. **Parameters** – Custom values to input when launching the stack.
5. **Mappings** – Hardcoded value lookups (e.g., region to AMI mapping).
6. **Conditions** – Logical conditions for creating resources (e.g., only in prod).
7. **Rules** – Validate parameter values (mostly used with AWS Service Catalog).
8. **Transform** – For macro-like processing (e.g., using AWS::Serverless transform). **Commonly used with AWS SAM (Serverless Application Model)** to simplify serverless app templates.
9. **Resources** – The actual AWS resources to create.
10. **Outputs** – Values to return after stack creation (e.g., BucketName, URLs).

📦 **Example: Create an S3 Bucket Template with All Sections**

yaml

```
AWSTemplateFormatVersion: "2010-09-09"

Description: >
```

```yaml
  CloudFormation Template to create an S3 bucket with all major sections
demonstrated.

Metadata:
  Author: Abhijit Gupta
  Version: "1.0"
  Purpose: Demo of all CloudFormation template sections

Parameters:
  BucketEnvironment:
    Type: String
    Description: Choose the environment
    AllowedValues: [dev, prod]
    Default: dev

  BucketName:
    Type: String
    Description: Name of the S3 bucket to create

Mappings:
  RegionMap:
    us-east-1:
      StorageClass: STANDARD
    ap-south-1:
      StorageClass: STANDARD_IA

Conditions:
  IsProd:
    Fn::Equals: [ !Ref BucketEnvironment, prod ]

Rules:
  BucketNameRule:
    Assertions:
      - Assert:
          Fn::Not:
            - Fn::Equals: [ !Ref BucketName, "" ]
        AssertDescription: Bucket name must not be empty.

Transform: AWS::Serverless-2016-10-31  # Not required here, but shown as example

Resources:
  MyS3Bucket:
    Type: AWS::S3::Bucket
    Condition: IsProd  # Only create this bucket if environment is 'prod'
    Properties:
      BucketName: !Ref BucketName
      Tags:
```

```
      - Key: Environment
        Value: !Ref BucketEnvironment
      - Key: StorageClass
        Value: !FindInMap [RegionMap, !Ref "AWS::Region", StorageClass]

Outputs:
  S3BucketName:
    Description: Name of the created S3 bucket
    Value: !Ref MyS3Bucket

  BucketARN:
    Description: ARN of the bucket
    Value: !GetAtt MyS3Bucket.Arn
```

What Does This Template Do?

- Accepts user input for BucketEnvironment and BucketName.
- Checks via **Rules** that BucketName is not empty.
- Uses **Mappings** to pick a storage class based on the region.
- Uses **Conditions** to create the bucket **only if environment is prod**.
- Applies **Metadata** to document template author and purpose.
- Uses a dummy **Transform** section (AWS::Serverless), typically used for SAM apps.
- Defines actual **Resources** (S3 bucket).
- Shows **Outputs** for reuse in other stacks or just to get bucket info.

Resources: Mandatory

Which section allows logical decisions like creating resources only for specific environments?
A. Metadata
B. Mappings
C. Conditions
D. Outputs
Answer: C

Which section is optional, not processed by CloudFormation, and is mainly for storing extra information?
A. Metadata
B. Resources
C. Description
D. Rules
Answer: A

What is the correct use of the Mappings section?
A. To create key-value outputs
B. To create S3 buckets

C. To define region-specific values like AMI IDs
D. To define macro transformations
Answer: C
Which section is primarily used to perform input validation, especially with AWS Service Catalog?
A. Parameters
B. Conditions
C. Rules
D. Metadata
Answer: C