Q1.Solution:

In [1]:
```
# Define the grade criteria
grade_criteria = {
    'A': (90, 100),
    'B': (80, 89),
    'C': (70, 79),
    'D': (60, 69),
    'F': (0, 59)
}

def get_grade(percentage):
    """Return the grade corresponding to the given percentage"""
    for grade, (lower, upper) in grade_criteria.items():
        if lower <= percentage <= upper:
            return grade
    return "Invalid percentage"

def main():
    # Get the percentage from the user
    while True:
        try:
            percentage = float(input("Enter your percentage: "))
            if 0 <= percentage <= 100:
                break
            else:
                print("Please  a percentage between 0 and 100.")
        except ValueError:
            print("Invalid input. Please enter a number.")

    # Display the grade
    grade = get_grade(percentage)
    print(f"Your grade is: {grade}")

if __name__ == "__main__":
    main()
```

```
Your grade is: B
```

In [2]:
```
#Q2.Solution
```

In [3]:
```
# Get the cost price of the bike from the user
cost_price = float(input("Enter the cost price of the bike: "))

# Calculate the road tax based on the cost price
if cost_price <= 10000:
    road_tax = cost_price * 0.02
elif 10000 < cost_price <= 20000:
    road_tax = cost_price * 0.04
elif 20000 < cost_price <= 50000:
    road_tax = cost_price * 0.06
else:
    road_tax = cost_price * 0.08

# Display the road tax to be paid
print("The road tax to be paid is: ₹", round(road_tax, 2))
```

```
The road tax to be paid is: ₹ 800.0
```

Q3.Solution

In [4]:
```python
# city_monuments.py

# Pre-defined dictionary of city-monument data
city_monuments = {
    "Paris": ["Eiffel Tower", "Louvre Museum", "Notre-Dame Cathedral"],
    "Rome": ["Colosseum", "Pantheon", "Trevi Fountain"],
    "New York City": ["Statue of Liberty", "Central Park", "Empire State Buildin
    "London": ["Buckingham Palace", "Tower of London", "Big Ben"],
    # Add more cities and monuments as needed...
}

def display_monuments(city):
    """Display monuments for a given city"""
    city = city.title()  # Normalize city name to title case
    if city in city_monuments:
        print(f"Monuments in {city}:")
        for monument in city_monuments[city]:
            print(f"* {monument}")
    else:
        print(f"Sorry, no monuments found for {city}.")

def main():
    city = input("Enter a city: ")
    display_monuments(city)

if __name__ == "__main__":
    main()
```

```
Sorry, no monuments found for Tower Of London.
```

Q4.Solution:

In [5]:
```python
def divide_by_three(n):
    count = 0
    while n > 10:
        n /= 3
        count += 1
    return count

# Example usage:
number = 1000
result = divide_by_three(number)
print(f"The number {number} can be divided by 3 {result} times before it is less
```

```
The number 1000 can be divided by 3 5 times before it is less than or equal to
10.
```

Q5 Solution:

Ans : In Python, a while loop is a control structure that allows you to execute a block of code repeatedly as long as a certain condition is met. It is a fundamental construct in programming that enables you to iterate over a sequence of statements until a specific goal is achieved.

When to Use a While Loop

You should use a while loop in the following situations:

Unknown number of iterations: When you don't know in advance how many times you need to repeat a block of code, a while loop is a good choice. Dependent on a condition: When the repetition of a block of code depends on a specific condition being met, a while loop is suitable. Manual iteration control: When you need to manually control the iteration process, such as incrementing or decrementing a counter, a while loop provides the necessary flexibility.

# Example 1: Simple While Loop

In [8]:
```
i = 1
while i <= 5:
    print(i)
    i += 1
```

```
1
2
3
4
5
```

# Example 2: Using a While Loop to Find the Sum of Numbers

In [9]:
```
i = 1
total = 0
while i <= 10:
    total += i
    i += 1
print("Sum:", total)
```

```
Sum: 55
```

Q6.Solution:

# Pattern 1: Right Triangle

In [10]:
```
i = 1
while i <= 5:
    j = 1
    while j <= i:
        print("*", end=" ")
        j += 1
    print()
    i += 1
```

```
*
* *
* * *
* * * *
* * * * *
```

# Pattern 2: Pyramid

In [11]:
```
i = 1
while i <= 5:
    j = 1
    while j <= 5 - i:
        print(" ", end=" ")
        j += 1
    k = 1
    while k <= 2 * i - 1:
        print("*", end=" ")
        k += 1
    print()
    i += 1
```

```
        *
      * * *
    * * * * *
  * * * * * * *
* * * * * * * * *
```

# Pattern 3: Diamond

In [12]:
```
i = 1
while i <= 5:
    j = 1
    while j <= 5 - i:
        print(" ", end=" ")
        j += 1
    k = 1
    while k <= 2 * i - 1:
        print("*", end=" ")
        k += 1
    print()
    i += 1
i = 4
while i >= 1:
    j = 1
    while j <= 5 - i:
        print(" ", end=" ")
        j += 1
    k = 1
    while k <= 2 * i - 1:
        print("*", end=" ")
        k += 1
    print()
    i -= 1
```

```
        *
      * * *
    * * * * *
  * * * * * * *
* * * * * * * * *
  * * * * * * *
    * * * * *
      * * *
        *
```

Q7.Solution:

In [13]:
```
i = 10
while i >= 1:
    print(i)
    i -= 1
```

```
10
9
8
7
6
5
4
3
2
1
```

In [ ]:
```

```