| Name of student: Abhay Omprakash Prajapati | | |
|---|---|---|
| Roll no: 41 | Tutorial No: 2 | |
| Title of LAB Assignment: To implement Python programs with conditionals and loops | | |
| DOP: 25-09-2023 | DOS:02-10-2023 | |
| CO Mapped: Co1,Co2 | PO Mapped: PO3 ,PO6 | Signature: |

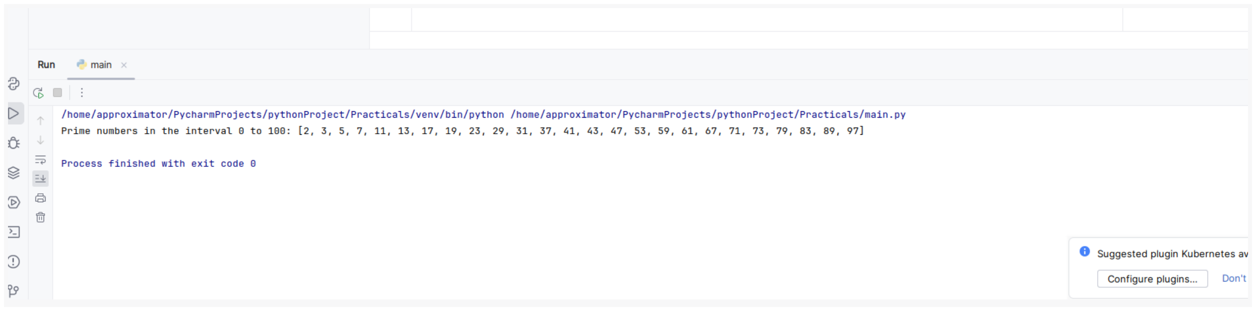
1. To find all the prime numbers in the interval 0 to 100

**Aim**: To find and display all prime numbers in the range from 0 to 100.
**Theory**: Prime numbers are natural numbers greater than 1 that have no positive divisors other than 1 and themselves. In this task, we'll iterate through numbers from 2 to 100 and check if each number is prime or not. We'll display all the prime numbers found in the given interval.
**Code:**
```
def is_prime(num):
    if num <= 1:
        return False
    for i in range(2, int(num ** 0.5) + 1):
```

```python
        if num % i == 0:
            return False
    return True

primes = [num for num in range(2, 101) if is_prime(num)]
print("Prime numbers in the interval 0 to 100:", primes)
```

**Conclusion**: We checked if the given number is an Armstrong number and provided the result.

**Output**:



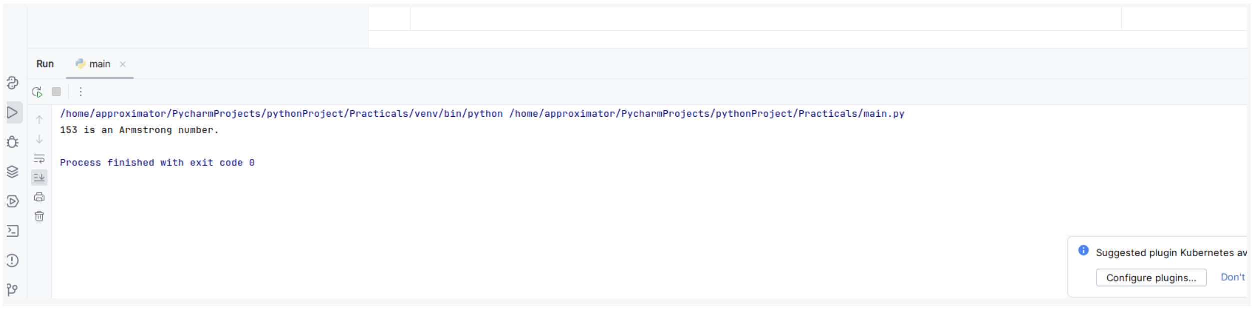
```
/home/approximator/PycharmProjects/pythonProject/Practicals/venv/bin/python /home/approximator/PycharmProjects/pythonProject/Practicals/main.py
Prime numbers in the interval 0 to 100: [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97]

Process finished with exit code 0
```

## 2. To check if the given number is Armstrong number or not

**Aim:** To check if a given number is an Armstrong number.
**Theory**: An Armstrong number (or narcissistic number) is a number that is equal to the sum of its own digits raised to the power of the number of digits. In this task, we'll check if a given number is an Armstrong number or not.
**Code**:

```python
def is_armstrong(num):
    num_str = str(num)
    num_digits = len(num_str)
    total = sum(int(digit) ** num_digits for digit in num_str)
    return num == total

number = 153
if is_armstrong(number):
    print(number, "is an Armstrong number.")
else:
    print(number, "is not an Armstrong number.")
```

**Conclusion:** We checked if the given number is an Armstrong number and provided the result.



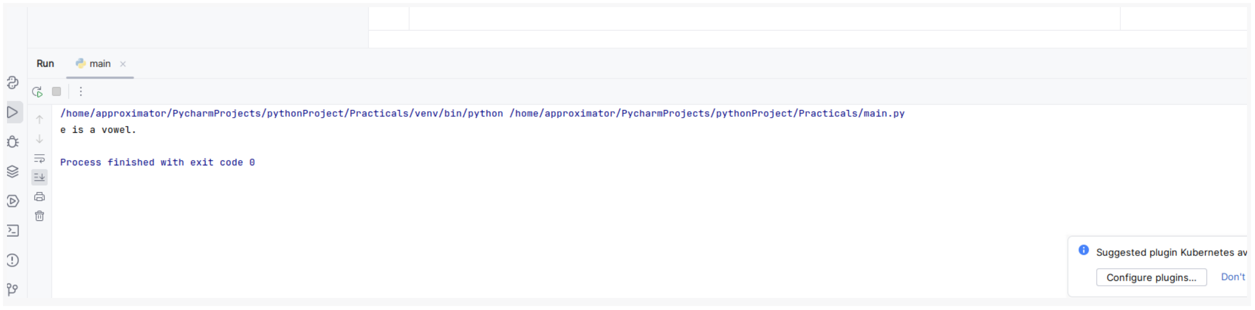
```
/home/approximator/PycharmProjects/pythonProject/Practicals/venv/bin/python /home/approximator/PycharmProjects/pythonProject/Practicals/main.py
153 is an Armstrong number.

Process finished with exit code 0
```

## 3. To check if the given character is a vowel or consonant

**Aim**: To check if a given character is a vowel or consonant.
**Theory:** Vowels are the letters 'a', 'e', 'i', 'o', and 'u'. In this task, we'll check if a given character is a vowel or a consonant.
**Code:**

```python
char = 'e'  # You can change this to any character you want to check
if char.lower() in ('a', 'e', 'i', 'o', 'u'):
   print(char, "is a vowel.")
else:
   print(char, "is a consonant.")
```

**Conclusion**: We checked if the given character is a vowel or a consonant and provided the result.



```
/home/approximator/PycharmProjects/pythonProject/Practicals/venv/bin/python /home/approximator/PycharmProjects/pythonProject/Practicals/main.py
e is a vowel.

Process finished with exit code 0
```

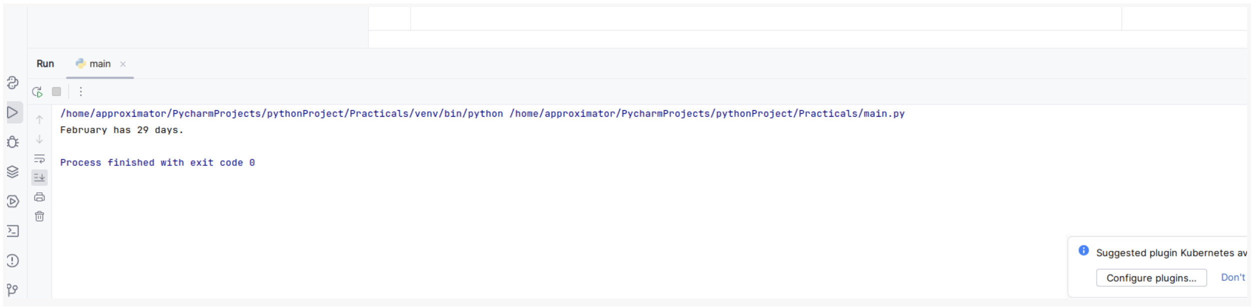
## 4. To convert a month to a number of days

**Aim:** To convert a month to the number of days it contains.
**Theory:** Different months have different numbers of days. In this task, we'll convert a given month to the number of days it contains, considering both common years and leap years.
**Code:**

```python
def month_to_days(month, is_leap_year=False):
    month = month.lower()
    if month in ("january", "march", "may", "july", "august", "october",
"december"):
        return 31
    elif month in ("april", "june", "september", "november"):
        return 30
    elif month == "february":
        return 29 if is_leap_year else 28
    else:
        return None  # Invalid month

given_month = "February"  # You can change this to any month you want to check
is_leap = True  # You can change this to False for a non-leap year
days = month_to_days(given_month, is_leap)
if days is not None:
    print(given_month, "has", days, "days.")
else:
    print("Invalid month.")
```

**Conclusion:** We converted a given month to the number of days it contains and provided the result.

**Output:**



```
/home/approximator/PycharmProjects/pythonProject/Practicals/venv/bin/python /home/approximator/PycharmProjects/pythonProject/Practicals/main.py
February has 29 days.

Process finished with exit code 0
```

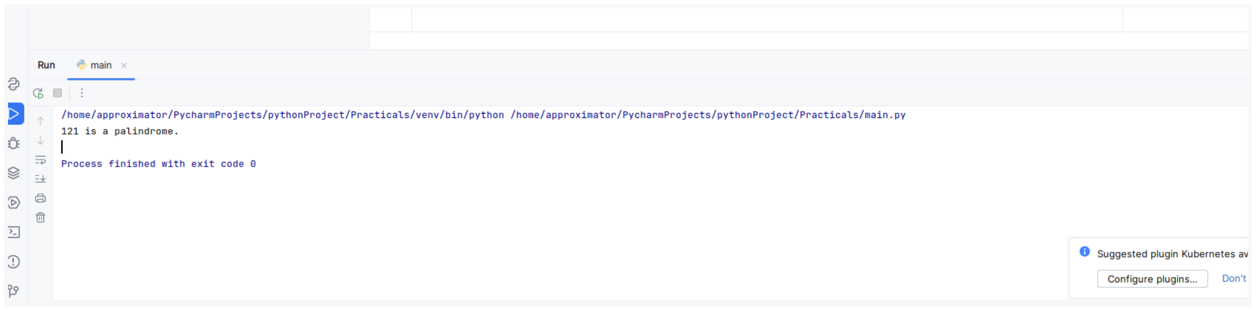
## 5. To check if a number is a palindrome or not

**Aim:** To check if a given number is a palindrome.
**Theory:** A palindrome is a number that remains the same when its digits are reversed. In this task, we'll check if a given number is a palindrome or not.
**Code:**

```python
def is_palindrome(number):
    num_str = str(number)
    return num_str == num_str[::-1]

given_number = 121  # You can change this to any number you want to check
if is_palindrome(given_number):
    print(given_number, "is a palindrome.")
else:
    print(given_number, "is not a palindrome.")
```

**Conclusion**: We checked if the given number is a palindrome and provided the result.

**Output:**



```
/home/approximator/PycharmProjects/pythonProject/Practicals/venv/bin/python /home/approximator/PycharmProjects/pythonProject/Practicals/main.py
121 is a palindrome.

Process finished with exit code 0
```

## 6. Program to Take in the Marks of 3 Subjects and Display the Grade

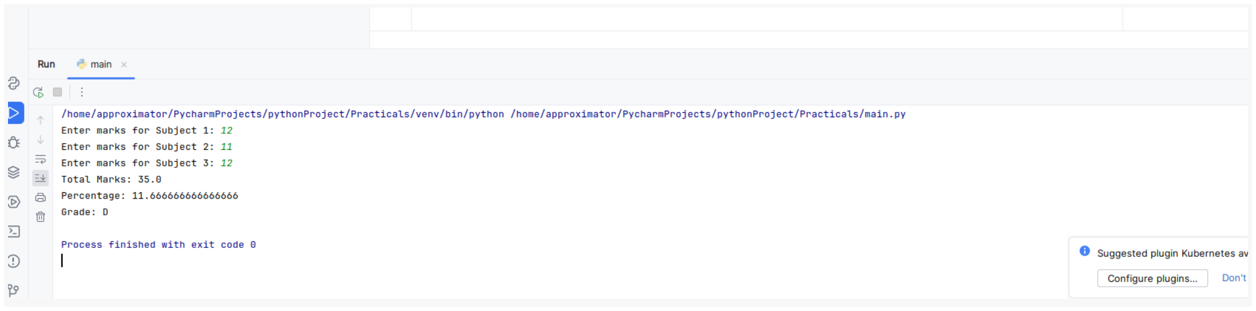
**Aim:** To calculate the grade based on the marks of 3 subjects.
**Theory:** In this task, we'll take marks for 3 subjects as input and then calculate the average percentage. Based on the percentage, we'll assign a grade.
**Code:**

```python
subject1 = float(input("Enter marks for Subject 1: "))
subject2 = float(input("Enter marks for Subject 2: "))
subject3 = float(input("Enter marks for Subject 3: "))

total_marks = subject1 + subject2 + subject3
percentage = (total_marks / 300) * 100

if percentage >= 90:
    grade = "A+"
elif percentage >= 80:
    grade = "A"
elif percentage >= 70:
    grade = "B"
elif percentage >= 60:
    grade = "C"
else:
    grade = "D"
```

```python
print("Total Marks:", total_marks)
print("Percentage:", percentage)
print("Grade:", grade)
```

**Conclusion:** We calculated the grade based on the marks of 3 subjects and displayed the result.

**Output:**



```
/home/approximator/PycharmProjects/pythonProject/Practicals/venv/bin/python /home/approximator/PycharmProjects/pythonProject/Practicals/main.py
Enter marks for Subject 1: 12
Enter marks for Subject 2: 11
Enter marks for Subject 3: 12
Total Marks: 35.0
Percentage: 11.666666666666666
Grade: D

Process finished with exit code 0
```

## 7. To add two matrices

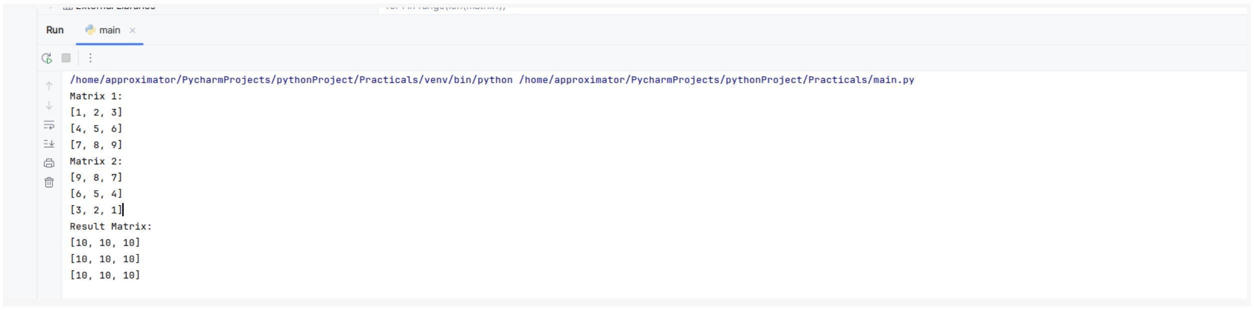
**Aim:** To add two matrices.

**Theory:** In this task, we'll add two matrices of the same dimensions. Matrix addition involves adding corresponding elements of two matrices to form a new matrix.

**Code:**

```python
matrix1 = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
matrix2 = [[9, 8, 7], [6, 5, 4], [3, 2, 1]]

result = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]

for i in range(len(matrix1)):
   for j in range(len(matrix1[0])):
       result[i][j] = matrix1[i][j] + matrix2[i][j]


print("Matrix 1:")
for row in matrix1:
   print(row)

print("Matrix 2:")
for row in matrix2:
   print(row)

print("Result Matrix:")
for row in result:
    print(row)
```

**Output:**



```
/home/approximator/PycharmProjects/pythonProject/Practicals/venv/bin/python /home/approximator/PycharmProjects/pythonProject/Practicals/main.py
Matrix 1:
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]
Matrix 2:
[9, 8, 7]
[6, 5, 4]
[3, 2, 1]
Result Matrix:
[10, 10, 10]
[10, 10, 10]
[10, 10, 10]
```

## 8. To check the validity of a password input by users

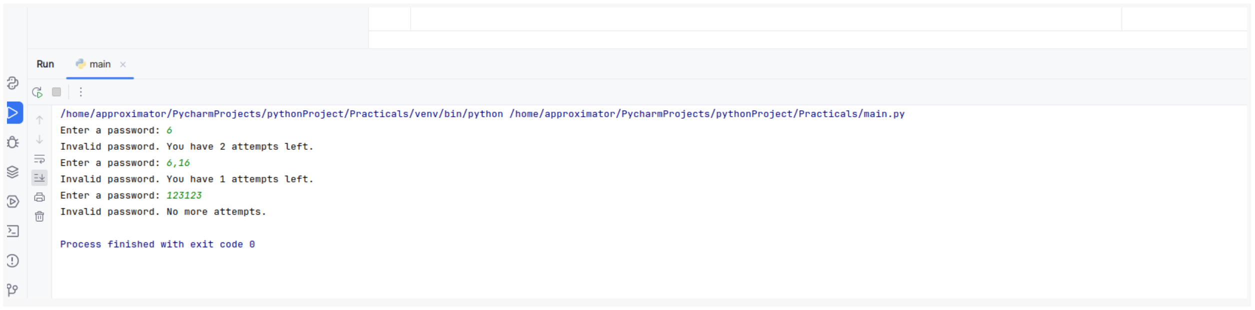
**Aim:** To check the validity of a user-entered password.
**Theory:** In this task, we'll validate a password based on the specified criteria. The password must contain at least 1 lowercase letter, 1 uppercase letter, 1 digit, and 1 special character from the set [$#@]. It should also have a minimum length of 6 characters and a maximum length of 16 characters. The user has 3 chances to enter a valid password.
Code:

**Code:**

```
import re

attempts = 3

while attempts > 0:
    password = input("Enter a password: ")
    if re.match(r"^(?=.*[a-z])(?=.*[A-Z])(?=.*[0-9])(?=.*[$#@]).{6,16}$", 
password):
        print("Password is valid.")
        break
    else:
        attempts -= 1
        if attempts > 0:
            print("Invalid password. You have", attempts, "attempts left.")
        else:
            print("Invalid password. No more attempts.")
```

**Conclusion**: We checked the validity of a user-entered password and provided the result within 3 chances.

**Output:**



```
/home/approximator/PycharmProjects/pythonProject/Practicals/venv/bin/python /home/approximator/PycharmProjects/pythonProject/Practicals/main.py
Enter a password: 6
Invalid password. You have 2 attempts left.
Enter a password: 6,16
Invalid password. You have 1 attempts left.
Enter a password: 123123
Invalid password. No more attempts.

Process finished with exit code 0
```