Q. Explain type conversion with example.

Ans: Implicit Type Conversion:

This occurs automatically when the data types are compatible, and there is no loss of information.

eg: Python converts num_int (an integer) to a float automatically before performing the addition

```
num_int = 10
num_float = num_int + 2.5
print(num_float)
print(type(num_float))
```

Output: 12.5

Output: <class 'float'>

Explicit Type Conversion:

This requires the programmer to use predefined functions like int(), float(), str(), etc., to convert data types.

eg: the programmer explicitly specifies the desired data type using conversion functions.

```
num_float = 5.7
num_int = int(num_float)
print(num_int)
print(type(num_int))
```

Output: 5

Output: <class 'int'>

Q. Write a program that accepts the lengths of three sides of a triangle as inputs and outputs whether or not the triangle is a right triangle.

Ans:

```
def is_right_triangle(a, b, c):
    sides = sorted([a, b, c])
    return sides[0]**2 + sides[1]**2 == sides[2]**2

try:
    side1 = float(input("Enter the length of the first side: "))
    side2 = float(input("Enter the length of the second side: "))
    side3 = float(input("Enter the length of the third side: "))

    if is_right_triangle(side1, side2, side3):
    print("The triangle is a right triangle.")
    else:
    print("The triangle is NOT a right triangle.")
except ValueError:
    print("Please enter valid numeric values for the sides.")
```

Output:

Enter the length of the first side: 3

Enter the length of the second side: 4
Enter the length of the third side: 5
The triangle is a right triangle.

Q. Write a Python program to print all numbers between 100 and 1000 whose sum of digits is divisible by 9.

Ans:
```
def sum_of_digits(num):
    return sum(int(digit) for digit in str(num))

print("Numbers between 100 and 1000 whose sum of digits is divisible by 9:")
for number in range(100, 1000):
    if sum_of_digits(number) % 9 == 0:
    print(number, end=" ")
```

Output:

Numbers between 100 and 1000 whose sum of digits is divisible by 9:

108 117 126 135 144 153 162 171 180 189 198 207 216 225 234 243 252 261 270 279 288 297 306 315 324 333 342 351 360 369 378 387 396 405 414 423 432 441 450 459 468 477 486 495 504 513 522 531 540 549 558 567 576 585 594 603 612 621 630 639 648 657 666 675 684 693 702 711 720 729 738 747 756 765 774 783 792 801 810 819 828 837 846 855 864 873 882 891 900 909 918 927 936 945 954 963 972 981 990 999

Q. Illustrate the use of range() in Python.

Ans: The range() function in Python generates a sequence of numbers. It is commonly used in loops to iterate over a sequence of values.

Syntax:

```
range(stop)
range(start, stop)
range(start, stop, step)
```

eg: Generate numbers from 1 to 10 with a step of 2

```
for i in range(1, 11, 2):
    print(i, end=" ")
```

Output: 1 3 5 7 9

Q. Write a Python program to print all prime factors of a given number.

Ans:
```
def prime_factors(n):
    print(f"Prime factors of {n} are:", end=" ")

    while n % 2 == 0:
    print(2, end=" ")
```

```
    n //= 2

    factor = 3
  while factor * factor <= n:
    while n % factor == 0:
                    print(factor, end=" ")
                    n //= factor
    factor += 2

    if n > 2:
    print(n, end=" ")
    print()



try:
    number = int(input("Enter a positive integer: "))
    if number > 0:
    prime_factors(number)
    else:
    print("Please enter a positive integer.")
except ValueError:
    print("Invalid input. Please enter a valid positive integer.")
```