**Q1 Create a Python program that uses "filter()" to remove all the vowels from a given string.**

def remove\_vowels(input\_string):

vowels = 'aeiouAEIOU'

**# Use filter to exclude vowels**

filtered\_chars = filter(lambda char: char not in vowels, input\_string)

**# Join the filtered characters back into a string**

return ''.join(filtered\_chars)

**# Example usage**

input\_string = "Hello, World!"

result = remove\_vowels(input\_string)

print(result) # Output: "Hll, Wrld!"

**Q2 Write a Python function that checks if a given number is prime or not from 1 to 200 ?**

def is\_prime(n):

    if n <= 1:

        return False

    for i in range(2, int(n\*\*0.5) + 1):

        if n % i == 0:

return False

    return True

# Test the function for numbers from 1 to 200

for num in range(1, 201):

    if is\_prime(num):

        print(f"{num} is a prime number.")

    else:

  print(f"{num} is not a prime number.")

**Q3 Create a Python function that accepts a string and returns the reverse of that string.**

def reverse\_string(s):

    return s[::-1]

**# Example usage**

result = reverse\_string("Hello, World!")

print(result)  # Output: !dlroW ,olleH

**Q4 Imagine an accounting routine used in a book shop. It works on a list with sublists, which look like this:**

**Order Number** **Book Title and author** **Quantity**  **price per item**

34587 learning python, Mark Lutz 4 40.95

98762 programming Python , mark Lutz 5 56.80

77226 Head First Python, Paul Barry 3 32.95

88112 Einfuhrung in Python3, Bernd Klein 3 24.99

orders = [

    {"order\_number": 34587, "book\_title": "Learning Python", "author": "Mark Lutz", "quantity": 4, "price\_per\_item": 40.95},

    {"order\_number": 98762, "book\_title": "Programming Python", "author": "Mark Lutz", "quantity": 5, "price\_per\_item": 56.80},

    {"order\_number": 77226, "book\_title": "Head First Python", "author": "Paul Barry", "quantity": 3, "price\_per\_item": 32.95},

    {"order\_number": 88112, "book\_title": "Einführung in Python3", "author": "Bernd Klein", "quantity": 3, "price\_per\_item": 24.99},

]

def calculate\_totals(orders):

    total\_order\_amount = 0

    summary\_report = []

    for order in orders:

        total\_price = order['quantity'] \* order['price\_per\_item']

        total\_order\_amount += total\_price

        summary\_report.append({

            "order\_number": order['order\_number'],

            "total\_price": total\_price

        })

  return summary\_report, total\_order\_amount

summary, overall\_total = calculate\_totals(orders)

**# Outputting the report**

for entry in summary:

    print(f"Order Number: {entry['order\_number']}, Total Price: {entry['total\_price']:.2f}")

print(f"Overall Total: {overall\_total:.2f}")

**5. Use a lambda function in Python to sort a list of tuples based on the second element of each tuple.**

# Define a list of tuples

data = [(1, 3), (2, 1), (3, 2), (4, 4)]

# Sort the list of tuples by the second element

sorted\_data = sorted(data, key=lambda x: x[1])

# Print the sorted result

print(sorted\_data)

**Q6 Write a Python program using lambda and map.**

# Define a list of integers

numbers = [1, 2, 3, 4, 5]

# Use map with a lambda function to square each number

squared\_numbers = list(map(lambda x: x \*\* 2, numbers))

# Print the result

print(squared\_numbers)

**Q7Implement a Python function that takes a list of integers and returns a new list containing the squares of each number.**

def square\_numbers(numbers):

return [x \*\* 2 for x in numbers]

# Example usage:

input\_list = [1, 2, 3, 4, 5]

squared\_list = square\_numbers(input\_list)

print(squared\_list)

**Q8 Write a generator function in Python that yields the powers of 2 up to a given exponent.**

def powers\_of\_two(max\_exponent):

for exponent in range(max\_exponent + 1):

yield 2 \*\* exponent

# Example usage:

for power in powers\_of\_two(5):

print(power