**Python Assignment**

Q1) Write a Python program to find the largest of three numbers.

Ans.1)  
a = int(input('Enter first number: '))  
b = int(input('Enter second number: '))  
c = int(input('Enter third number: '))  
print('Largest is:', max(a, b, c))

Q2) Create a program that checks whether a number is even or odd.

Ans.2)  
n = int(input('Enter a number: '))  
print('Even' if n % 2 == 0 else 'Odd')

Q3) Accept a string from the user and reverse it.

Ans.3)  
s = input('Enter a string: ')  
print('Reversed:', s[::-1])

Q4) Write a program to count the number of vowels in a string.

Ans.4)  
s = input('Enter a string: ')  
vowels = 'aeiouAEIOU'  
count = sum(1 for ch in s if ch in vowels)  
print('Vowel count:', count)

Q5) Create a calculator using if-else (add, subtract, multiply, divide).

Ans.5)  
a = float(input('Enter first number: '))  
b = float(input('Enter second number: '))  
op = input('Enter operator (+,-,\*,/): ')  
if op == '+': print(a + b)  
elif op == '-': print(a - b)  
elif op == '\*': print(a \* b)  
elif op == '/': print(a / b)  
else: print('Invalid operator')

Q6) Write a program to check if a number is prime.

Ans.6)  
n = int(input('Enter a number: '))  
if n < 2: print('Not Prime')  
else:  
 for i in range(2, int(n\*\*0.5)+1):  
 if n % i == 0:  
 print('Not Prime')  
 break  
 else:  
 print('Prime')

Q7) Display the multiplication table of a given number.

Ans.7)  
n = int(input('Enter a number: '))  
for i in range(1, 11):  
 print(f'{n} x {i} = {n\*i}')

Q8) Write a program to print the Fibonacci series up to n terms.

Ans.8)  
n = int(input('Enter number of terms: '))  
a, b = 0, 1  
for \_ in range(n):  
 print(a, end=' ')  
 a, b = b, a + b

Q9) Write a program to find the factorial of a number using a loop.

Ans.9)  
n = int(input('Enter a number: '))  
fact = 1  
for i in range(1, n+1):  
 fact \*= i  
print('Factorial:', fact)

Q10) Accept a number and check whether it is a palindrome.

Ans.10)  
n = input('Enter a number: ')  
print('Palindrome' if n == n[::-1] else 'Not Palindrome')

Q11) Find the sum of digits of a number using a while loop.

Ans.11)  
n = int(input('Enter a number: '))  
sum\_digits = 0  
while n > 0:  
 sum\_digits += n % 10  
 n //= 10  
print('Sum of digits:', sum\_digits)

Q12) Create a program to convert Celsius to Fahrenheit.

Ans.12)  
c = float(input('Enter Celsius: '))  
f = (c \* 9/5) + 32  
print('Fahrenheit:', f)

Q13) Write a Python function to find the maximum of two numbers.

Ans.13)  
def max\_two(a, b): return a if a > b else b  
print(max\_two(5, 10))

Q14) Accept a list of numbers and return the average.

Ans.14)  
lst = list(map(int, input('Enter numbers: ').split()))  
print('Average:', sum(lst)/len(lst))

Q15) Find the second largest number in a list.

Ans.15)  
lst = list(map(int, input('Enter numbers: ').split()))  
lst.remove(max(lst))  
print('Second largest:', max(lst))

Q16) Accept a sentence and count the number of words.

Ans.16)  
s = input('Enter a sentence: ')  
print('Word count:', len(s.split()))

Q17) Accept a list and remove all duplicates from it.

Ans.17)  
lst = list(map(int, input('Enter numbers: ').split()))  
print('Unique:', list(set(lst)))

Q18) Write a program to create a dictionary from two lists.

Ans.18)  
keys = ['a', 'b', 'c']  
values = [1, 2, 3]  
d = dict(zip(keys, values))  
print(d)

Q19) Accept a list and sort it without using sort() function.

Ans.19)  
lst = list(map(int, input('Enter numbers: ').split()))  
for i in range(len(lst)):  
 for j in range(i+1, len(lst)):  
 if lst[i] > lst[j]: lst[i], lst[j] = lst[j], lst[i]  
print('Sorted:', lst)

Q20) Write a function that checks if a string is a pangram.

Ans.20)  
import string  
def is\_pangram(s):  
 return set(string.ascii\_lowercase).issubset(s.lower())  
print(is\_pangram('The quick brown fox jumps over the lazy dog'))

Q21) Write a function to check if the input is a number or not.

Ans.21)  
def is\_number(s):  
 try:  
 float(s)  
 return True  
 except:  
 return False  
print(is\_number('123'))

Q22) Accept a string and check if it is an anagram of another.

Ans.22)  
from collections import Counter  
s1 = input('Enter first string: ')  
s2 = input('Enter second string: ')  
print('Anagram' if Counter(s1) == Counter(s2) else 'Not')

Q23) Accept a number and print its binary, octal, and hexadecimal.

Ans.23)  
n = int(input('Enter number: '))  
print('Binary:', bin(n))  
print('Octal:', oct(n))  
print('Hex:', hex(n))

Q24) Accept a list and print all elements greater than 50.

Ans.24)  
lst = list(map(int, input('Enter numbers: ').split()))  
print([x for x in lst if x > 50])

Q25) Write a function to count uppercase and lowercase letters in a string.

Ans.25)  
s = input('Enter string: ')  
uc = sum(1 for c in s if c.isupper())  
lc = sum(1 for c in s if c.islower())  
print('Upper:', uc, 'Lower:', lc)

Q26) Write a Python program using a lambda function to square all elements in a list.

Ans.26)  
lst = [1, 2, 3, 4]  
squares = list(map(lambda x: x\*\*2, lst))  
print(squares)

Q27) Create a function that returns a list of prime numbers from 1 to 100.

Ans.27)  
def is\_prime(n):  
 return n > 1 and all(n % i != 0 for i in range(2, int(n\*\*0.5)+1))  
print([x for x in range(1, 101) if is\_prime(x)])

Q28) Write a program to implement a simple calculator using functions.

Ans.28)  
def calc(a, b, op):  
 if op == '+': return a + b  
 if op == '-': return a - b  
 if op == '\*': return a \* b  
 if op == '/': return a / b  
print(calc(10, 2, '+'))

Q29) Implement a recursive function to compute factorial.

Ans.29)  
def factorial(n): return 1 if n==0 else n \* factorial(n-1)  
print(factorial(5))

Q30) Create a Python program using OOP for student management (class, object, init).

Ans.30)  
class Student:  
 def \_\_init\_\_(self, name, age):  
 self.name = name  
 self.age = age  
s = Student('Alice', 20)  
print(s.name, s.age)

Q31) Write a program to read a file and count the number of lines and words.

Ans.31)  
with open('sample.txt') as f:  
 lines = f.readlines()  
 print('Lines:', len(lines))  
 print('Words:', sum(len(line.split()) for line in lines))

Q32) Write a Python decorator to print the execution time of a function.

Ans.32)  
import time  
  
def timer(func):  
 def wrapper():  
 start = time.time()  
 func()  
 print('Time:', time.time() - start)  
 return wrapper  
  
@timer  
def test(): print('Running')  
test()

Q33) Create a class BankAccount with deposit and withdraw methods.

Ans.33)  
class BankAccount:  
 def \_\_init\_\_(self): self.balance = 0  
 def deposit(self, amount): self.balance += amount  
 def withdraw(self, amount): self.balance -= amount

Q34) Implement a generator to yield even numbers from 1 to n.

Ans.34)  
def even\_gen(n):  
 for i in range(2, n+1, 2): yield i  
print(list(even\_gen(10)))

Q35) Write a function using \*args and \*\*kwargs to print any number of arguments.

Ans.35)  
def print\_args(\*args, \*\*kwargs):  
 print('Args:', args)  
 print('Kwargs:', kwargs)  
print\_args(1,2,3, a=4,b=5)

Q36) Use list comprehension to create a list of squares of even numbers from 1 to 20.

Ans.36)  
print([x\*\*2 for x in range(1, 21) if x % 2 == 0])

Q37) Create a function that takes a list and returns only unique elements.

Ans.37)  
def unique(lst): return list(set(lst))  
print(unique([1,2,2,3]))

Q38) Write a Python program to merge two dictionaries.

Ans.38)  
d1 = {'a': 1, 'b': 2}  
d2 = {'c': 3}  
d1.update(d2)  
print(d1)

Q39) Create a class that inherits from another class and overrides a method.

Ans.39)  
class A:  
 def greet(self): print('Hello from A')  
class B(A):  
 def greet(self): print('Hello from B')  
b = B()  
b.greet()

Q40) Write a function to find common elements between two lists.

Ans.40)  
def common(lst1, lst2): return list(set(lst1) & set(lst2))  
print(common([1,2,3], [2,3,4]))

Q41) Implement a program to demonstrate try-except-finally for exception handling.

Ans.41)  
try:  
 x = int(input('Enter a number: '))  
except:  
 print('Invalid input')  
finally:  
 print('Done')

Q42) Write a program to parse and display JSON data from a string.

Ans.42)  
import json  
data = '{"name": "John", "age": 30}'  
parsed = json.loads(data)  
print(parsed)

Q43) Create a class that demonstrates the use of private and protected members.

Ans.43)  
class Demo:  
 def \_\_init\_\_(self):  
 self.\_protected = 1  
 self.\_\_private = 2  
d = Demo()  
print(d.\_protected)  
# print(d.\_\_private) # Error

Q44) Write a Python function to check if a string is a valid email address.

Ans.44)  
import re  
def is\_email(s):  
 return re.match(r'[^@]+@[^@]+\.[^@]+', s)  
print(is\_email('test@example.com'))

Q45) Accept a CSV file and convert it into a list of dictionaries.

Ans.45)  
import csv  
with open('data.csv') as f:  
 reader = csv.DictReader(f)  
 print(list(reader))

Q46) Create a simple login system using dictionary and input.

Ans.46)  
users = {'admin': '1234'}  
uname = input('Username: ')  
pwd = input('Password: ')  
print('Login' if users.get(uname) == pwd else 'Failed')

Q47) Write a function to flatten a nested list using recursion.

Ans.47)  
def flatten(lst):  
 for i in lst:  
 if isinstance(i, list): yield from flatten(i)  
 else: yield i  
print(list(flatten([1,[2,3],[4,[5]]])))

Q48) Demonstrate use of map(), filter(), and reduce() on a list.

Ans.48)  
from functools import reduce  
lst = [1,2,3,4]  
print('Map:', list(map(lambda x:x\*2, lst)))  
print('Filter:', list(filter(lambda x:x%2==0, lst)))  
print('Reduce:', reduce(lambda x,y:x+y, lst))

Q49) Build a small CLI app that takes user input and performs basic operations.

Ans.49)  
while True:  
 op = input('Enter operation (+,-,\*,/ or q): ')  
 if op == 'q': break  
 a = float(input('A: '))  
 b = float(input('B: '))  
 if op == '+': print(a + b)  
 elif op == '-': print(a - b)  
 elif op == '\*': print(a \* b)  
 elif op == '/': print(a / b)

Q50) Create a mini Python quiz that takes answers from the user and displays score.

Ans.50)  
questions = {'What is 2+2?': '4', 'What is capital of India?': 'Delhi'}  
score = 0  
for q, a in questions.items():  
 ans = input(q)  
 if ans.strip().lower() == a.lower(): score += 1  
print('Score:', score)