Deep Learning Lab 1

Selvakumar G (22MAI1004)

Program 1 : Get the following input from the user: first name and last name. Write a Python program to print as

Hello first name last name
Welcome to Python!

Code:

```
name = input("Enter your Full Name: ")
names = name.split(" ")
print(f"Hello {names[0]} {names[1]}\n Welcome to Python!")
```

Output:

```
/usr/local/bin/python3 /Users/selva/Documents/
selva@Selvakumars-MacBook-Pro lab % /usr/local
Enter your Full Name: Selvakumar G
Hello Selvakumar G
 Welcome to Python!
selva@Selvakumars-MacBook-Pro lab %
```

Program 2: Assign x = 8 and y = 2. Evaluate the following expressions and write the output,
a) x + y * 3
b) (x + y) * 3
c) x ** y
d) x % y

Code:

```
x, y = 8, 2
print(f"x+y*3 = {x+y*3}\n(x+y)*3 = {(x+y)*3}\nx**y = {x**y}\nx%y = {x%y}")
```

Output:

```
/usr/local/bin/python3 /Users/selva/Documents/SEM-2/DL/lab/2.py
selva@Selvakumars-MacBook-Pro lab % /usr/local/bin/python3 /User
x+y*3 = 14
(x+y)*3 = 30
x**y = 64
x%y = 0
selva@Selvakumars-MacBook-Pro lab %
```

Program 3: An employee's total weekly pay equals the hourly wage multiplied by the total number of regular hours plus any overtime pay. Overtime pay equals the total overtime hours multiplied by 1.5 times the hourly wage. Write a program that takes as inputs the hourly wage, total regular hours, and total overtime hours and displays an employee's total weekly pay.

Code:

```
hourly_wage = float(input("Enter the Hourly Wage of the Employ: "))
total_regular_hours = float(input("Enter the total regular Hours: "))
total_overtime_hours = float(input("Enter the total overtime Hours: "))

overtime_pay = total_overtime_hours * (1.5 * hourly_wage)

total_weekly_pay = hourly_wage * total_regular_hours + overtime_pay

print(total_weekly_pay)
```

Output:

```
selva@Selvakumars-MacBook-Pro lab % /usr/local/bin/python3 /Use
Enter the Hourly Wage of the Employ: 12.5
Enter the total regular Hours: 8
Enter the total overtime Hours: 4
175.0
selva@Selvakumars-MacBook-Pro lab %
```

Problem 4: A company decides to give bonus to all its employees for Diwali. A 5% bonus on salary is given to the male workers and 15% bonus on salary to the female workers. If the salary of the employee is less than Rs. 10000/- then the employee gets an extra 2% bonus on salary. Calculate the bonus that the employee will get and display the total salary.

Code:

```
gender = input("Enter the Employee Gender (M for Male, F for Female) : ")
salary = int(input("Enter the Employee Salary"))

total_salary = salary

if salary < 10000:
    bonus = salary * 0.02
    total_salary += bonus

if gender == 'm':
    bonus = salary * 0.05
    total_salary += bonus
elif gender == "f":
    bonus = salary * 0.15
    total_salary += bonus

print(bonus)
print(total_salary)
```

Output:

```
selva@Selvakumars-MacBook-Pro lab % /usr/local/bin/python3 /User
Enter the Employee Gender (M for Male, F for Female) : f
Enter the Employee Salary10000
1500.0
11500.0
selva@Selvakumars-MacBook-Pro lab %
```

Problem 5: Write a series of Python statements that will read three strings from the user, and then print them in dictionary order. (Note: you can compare two strings using the relational operators).

Code:

```
str1 = input("Enter a string: ")
str2 = input("Enter a string: ")
str3 = input("Enter a string: ")


if str1 < str2 and str1 < str3:
    if str2 < str3:
        print(str1, str2, str3)
    else:
        print(str1, str3, str2)

elif str2 < str1 and str2 < str3:
    if str1 < str3:
        print(str2, str1, str3)
    else:
        print(str2, str3, str1)

else:
    if str1 < str2:
        print(str3, str1, str2)
    else:
        print(str3, str2, str1)
```

Output:

```
selva@Selvakumars-MacBook-Pro lab % /usr/local/bin/python3 /Users
Enter a string: selva
Enter a string: kumar
Enter a string: apple
apple kumar selva
selva@Selvakumars-MacBook-Pro lab %
```

Program 6: Write a program that takes user's name and PAN card number. Validate the information using string functions

Code:

```
name = input("Enter the user name: ")
pan_no = input("Enter the PAN number: ")

print("Checking whether the name is valid: ", name.isalpha())
print("Checking whether the PAN number is valid: ", pan_no.isalnum())
```

Output:

```
selva@Selvakumars-MacBook-Pro lab % /usr/local/bin/python3 /User
Enter the user name: Selva
Enter the PAN number: abc123zx
Checking whether the name is valid:  True
Checking whether the PAN number is valid:  True
selva@Selvakumars-MacBook-Pro lab %
```

Program 7: Write a Python program to parse an email id to print from which email server it was sent.

Code:

```
mail = input("Enter the Mail Id: ")
print(mail.split("@")[1])
```

Output:

```
selva@Selvakumars-MacBook-Pro lab % /usr/local/bin/python3 /Us
Enter the Mail Id: iselvakumarg@gmail.com
gmail.com
selva@Selvakumars-MacBook-Pro lab %
```

Program 8: Write a Python program to strip a set of characters from a string Encrypt a given message by “rotating” each letter by a fixed number of places. To rotate a letter means to shift it through the alphabet, wrapping around to the beginning if necessary, so ‘A’ rotated by 3 is ‘D’ and ‘Z’ rotated by 1 is ‘A’. Write a function called rotate_word that takes a string and an integer as parameters, and returns a new string that contains the letters from the original string rotated by the given amount. E.g Given String: HAL Encrypted String: JCN (Rotated by 2)

Code:

```
def rotate_word(s, n):
    return ''.join([chr(ord(i) + n) for i in s])

print(rotate_word('HAL', 2))
```

Output:

```
/usr/local/bin/python3 /Users/selva/Document
selva@Selvakumars-MacBook-Pro lab % /usr/loc
JCN
selva@Selvakumars-MacBook-Pro lab %
```

Program 9: Ackermann Function

Code:

```
def ack(m, n):
    if m == 0:
        return n + 1
    elif m > 0 and n == 0:
        return ack(m - 1, 1)
    elif m > 0 and n > 0:
        return ack(m - 1, ack(m, n - 1))

print(ack(3, 4))
```

Output:

```
selva@Selvakumars-MacBook-Pro lab % /usr/loc
125
selva@Selvakumars-MacBook-Pro lab %
```

Program 10: Given a list, find frequency of each element and save it as list of tuple [(number, frequency)]. Input : test_list = [4, 5, 4, 5, 6, 6, 5] Output : [(4, 2), (5, 3), (6, 2)] Input : test_list = [4, 5, 4, 5, 6, 6, 6] Output : [(4, 2), (5, 3), (6, 3)]

Code:

```
def frequency(l):
    return [(i, l.count(i)) for i in set(l)]

print(frequency([4, 5, 4, 5, 6, 6, 5]))
print(frequency([4, 5, 4, 5, 6, 6, 6]))
```

Output:

```
selva@Selvakumars-MacBook-Pro lab % /usr/lo
[(4, 2), (5, 3), (6, 2)]
[(4, 2), (5, 2), (6, 3)]
selva@Selvakumars-MacBook-Pro lab %
```

Program 11: Remove duplicates from the tuple

Code:

```
def remove_duplicates(t):
    return tuple(set(t))

print(remove_duplicates((1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1, 2, 3, 4, 5, 6, 7, 8, 9,
10)))
```

Output:

```
selva@Selvakumars-MacBook-Pro lab % /usr/lo
(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
selva@Selvakumars-MacBook-Pro lab %
```

Program 12: Give an appropriate list comprehension for each of the following: i) L1 = [1, ’x’, 4, 5.6, ’z’, 9, ‘a’, 0, 4] create a list which consists of integer values. ii) Producing a list of consonants that appear in string w. iii) Multiples of 10 (n values) iv) Construct a list of the form: [‘1a’,’2a’,’3a’,’4a’] v) Create a list which stores the sum of each of the elements from the two lists.

```
# i) L1 = [1, ’x’, 4, 5.6, ’z’, 9, ‘a’, 0, 4] create a list which consists of
integer values.
L1 = [1, 'x', 4, 5.6, 'z', 9, 'a', 0, 4]
L2 = [i for i in L1 if type(i) == int]
print(L2)

# ii) Producing a list of consonants that appear in string w.
w = 'This is a string'
L3 = [i for i in w if i not in 'aeiou']
print(L3)

# iii) Multiples of 10 (n values)
L4 = [i for i in range(100) if i % 10 == 0]
print(L4)

# iv) Construct a list of the form: [‘1a’,’2a’,’3a’,’4a’]
L5 = [str(i) + 'a' for i in range(1, 5)]
print(L5)

# v) Create a list which stores the sum of each of the elements from the two lists.
L6 = [1, 2, 3, 4]
L7 = [5, 6, 7, 8]
L8 = [L6[i] + L7[i] for i in range(len(L6))]
print(L8)
```

Output

```
selva@Selvakumars-MacBook-Pro lab % /usr/lo
[1, 4, 9, 0, 4]
['T', 'h', 's', ' ', 's', ' ', ' ', 's', 't
[0, 10, 20, 30, 40, 50, 60, 70, 80, 90]
['1a', '2a', '3a', '4a']
[6, 8, 10, 12]
selva@Selvakumars-MacBook-Pro lab %
```

Problem 13: Obtain a nested list use looping constructs and list comprehension to flatten the list. Example: [[1],[2,3],[4,5,6],[7,8,9,10]] o/p: [1,2,3,4,5,6,7,8,9,10]

Code:

```
def flatten_list(l):
    return [i for j in l for i in j]

print(flatten_list([[1],[2,3],[4,5,6],[7,8,9,10]]))
```

Output:

```
  [6, 8, 10, 12]
● selva@Selvakumars-MacBook-Pro lab % /usr/loc
  [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
○ selva@Selvakumars-MacBook-Pro lab %
```

Problem 14: Create a dictionary for 6 employee details with empno as key, name, dob and net-pay as list of values use appropriate dictionary methods: a. Create a dictionary with the above information. b. Insert a new employee details as the second employee c. Delete the employee at the 4th position d. Delete the last employee e. Increment the salary of all employees by 5%.

Code:

```
#a. Create a dictionary with the above information.
employee_dict = {
     101: ["A", "1990-11-25", 70000],
     102: ["B", "1991-05-21", 10000],
     103: ["C", "1992-11-03", 35000],
     104: ["D", "1993-09-27", 85000],
     105: ["E", "1984-06-28", 80000],
     106: ["F", "1993-03-25", 15000]
}
# b. Insert a new employee details as the second employee
employee_dict[102] = ["G", "1994-08-12", 50000]
# C. Delete the employee at the 4th position
employee_dict.pop(104)
# d. Delete the last employee
employee_dict.popitem()
# e. Increment the salary of all employees by 5%
for emp_details in employee_dict. values():
    emp_details[2] *= 1.05

print(employee_dict,end="\n")
```

Output:

```
selva@Selvakumars-MacBook-Pro lab % /usr/local/bin/python3 /Users/selva/Documents/SEM-2/DL/lab/14.py
{101: ['A', '1990-11-25', 73500.0], 102: ['G', '1994-08-12', 52500.0], 103: ['C', '1992-11-03', 36750.0], 105: ['E', '1984-06-28', 84000.0]}
```

Problem 15: Consider the student details are maintained using nested dictionary as follows: {Reg no: {subcode: CAT1, CAT2, SAT}} a. Create nested dictionary for three subjects b. Display the information of a student given his register number c. To display the marks of a student given his subject code d. Update the details of the student given the register number.

Code:

```
students = {
    "101": {
         "sub1": [50, 60, 75],
         "sub2": [70, 80, 90],
         "sub3": [85, 90, 95]
     },
     "102": {
         "sub1": [60, 75, 80],
         "sub2": [80, 85, 90],
         "sub3": [90, 95, 98]
     },
     "103": {
         "subl": [70, 80, 85],
         "sub2": [85, 90, 92],
         "sub3": [95, 98, 99]
     }
}
#b. Displaying the information of a student given their register number:
def display_student_info(reg_no):
    if reg_no in students:
        print(f"Register number: {reg_no}")
        for sub_code, marks in students[reg_no].items():
            print(f"{sub_code} marks: {marks}")
    else:
        print(f"No student found with register number {reg_no}")

# c. Displaying the marks of a student in a particular subject given their register
number and subject code:
def display_marks_by_sub_code(reg_no, sub_code):
     if reg_no in students and sub_code in students[reg_no]:
        print(f"{sub_code} marks for student {reg_no}:
{students[reg_no][sub_code]}")
     else:
        print(f"No marks found for subject {sub_code} and student {reg_no}")

# d. Updating the details of a student given their register number, subject code
and marks:
def update_student_details(reg_no, sub_code, cat1, cat2, sat):
    if reg_no in students:
        if sub_code in students[reg_no]:
            students [reg_no][sub_code] = [cat1, cat2, sat]
            print(f"Updated {sub_code} marks for student {reg_no}:
{students[reg_no][sub_code]}")
            print(f"No subject {sub_code} found for student {reg_no}")
```

```
    else:
        print(f"No student found with register number {reg_no}")


print(display_student_info( "101"))
print(display_marks_by_sub_code("101", "sub1"))
print(update_student_details("101", "sub1", 70,70,70))
```

Output:

```
selva@Selvakumars-MacBook-Pro lab % /usr/local/bin/
Register number: 101
sub1 marks: [50, 60, 75]
sub2 marks: [70, 80, 90]
sub3 marks: [85, 90, 95]
None
sub1 marks for student 101: [50, 60, 75]
None
Updated sub1 marks for student 101: [70, 70, 70]
No subject sub1 found for student 101
None
selva@Selvakumars-MacBook-Pro lab %
```

Problem 16: Write a Python code to read the content of 'ebook.txt' and display the contents of the file onto the console.

ebook.txt

ebook.txt

```
1   Harry Potter, a boy who learns on his eleventh birthday that
    he is the orphaned son of two powerful wizards and possesses
    unique magical powers of his own. He is summoned from his
    life as an unwanted child to become a student at Hogwarts, an
    English boarding school for wizards. There, he meets several
    friends who become his closest allies and help him discover
    the truth about his parents' mysterious deaths.
```

Code:

```
with open("ebook.txt", "r") as f:
    print(f.read())
```

Output:

```
selva@Selvakumars-MacBook-Pro lab % /usr/local/bin/python3 /Users/selva/Documents/SEM-2/DL/lab/16.py
Harry Potter, a boy who learns on his eleventh birthday that he is the orphaned son of two powerful wizards and possesses unique magical powers of his own. He
 is summoned from his life as an unwanted child to become a student at Hogwarts, an English boarding school for wizards. There, he meets several friends who b
ecome his closest allies and help him discover the truth about his parents' mysterious deaths.
selva@Selvakumars-MacBook-Pro lab %
```

Problem 17: Write a function ‘display_words()’ in python to read lines from a text file "ebook.txt", and returns a list with words less than 4 characters.

Code:

```
def display_words():

    with open("ebook.txt", "r") as f:
        words = f.read().split()
        return [word for word in words if len(word) < 4]

print(display_words())
```

Output:

```
selva@Selvakumars-MacBook-Pro lab % /usr/local/bin/python3 /Users/selva/Documents/SEM-2/DL/lab/17.py
['a', 'boy', 'who', 'on', 'his', 'he', 'is', 'the', 'son', 'of', 'two', 'and', 'of', 'his', 'He', 'is', 'his', 'as', 'an', '
to', 'a', 'at', 'an', 'for', 'he', 'who', 'his', 'and', 'him', 'the', 'his']
selva@Selvakumars-MacBook-Pro lab %
```

Problem 18: Create a CSV file and store student details like Name, Register number and CGPA. Write a python code to read the file content and display the content. Duplicate entries and null value entries has to be ignored while displaying.

Code:

```
# Create a CSV file and store student details like Name, Register number and CGPA.
Write a python code to read the file content and display the content. Duplicate
entries and null value entries has to be ignored while displaying.

import csv
# Sample data
data = [
     ['Selva', '1001', '9.1'],
     ['Shiva', '1002', '9.1'],
     ['Nandhini', '1003', '7.8' ],
     ['Sri', '1004', '8.9' ],
     ['pawan', '1005' , '9.5'],
     ['Bavani', '1006', '8.5'],
     ['', '1007', '9.1'],
     ['shankar', '', '7.8' ]
]

with open('student.csv', mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['Name', 'Reg No', 'CGPA'])
    for row in data:
         if row[0] and row[1] and row[2]:
            writer.writerow(row)

with open( 'student.csv', mode= 'r') as file:
    reader = csv. reader(file)
    next(reader)
    for row in reader:
         if len(row) == 3 and row[0] and row[1] and row[2]:
```

```
            print(f"Name: {row[0]}, Reg No: {row[1]}, CGPA: {row[2]}")
```

Output:

```
selva@Selvakumars-MacBook-Pro lab % /usr/local/bi
Name: Selva, Reg No: 1001, CGPA: 9.1
Name: Shiva, Reg No: 1002, CGPA: 9.1
Name: Nandhini, Reg No: 1003, CGPA: 7.8
Name: Sri, Reg No: 1004, CGPA: 8.9
Name: pawan, Reg No: 1005, CGPA: 9.5
Name: Bavani, Reg No: 1006, CGPA: 8.5
selva@Selvakumars-MacBook-Pro lab %
```