

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Exp No**. | **Laboratory Experments Description** | **Page No.** |
| **CSCP409 : Data Structures Laboratory** |
| 1. | Write a python program to print the given pattern  a) 1 b) 1 2 3 4 5 c) \* \* \* \* d) 1  2 2 1 2 3 4 5 \* \* \* 3 2  3 3 3 1 2 3 4 5 \* \* 6 5 4  4 4 4 1 2 3 4 5 \* 10 9 8 7  1 2 3 4 5 | 1-3 |
| 2. | Write a python function to find GCD of 2 numbers | 4 |
| 3. | Write a python program to check whether the given string/number is palindrome or not? | 5 |
| 4. | Write a python program to print factorial of number | 6 |
| 5. | Write a python function to check whether the number is even or odd | 7 |
| 6. | A school has following rules for grading system  a)Below 25 is F  b)25 To 45 is E  c)45 to 50 is D  d)50 to 60 is C  e)60 to 80 is B  f)above 80 is A  Ask user to enter marks and print the corresponding grade | 8 |
| 7. | Write a program to produce the Fibonacci series | 9 |
| 8. | Write a python program to accept number from 1 to 7 and display the corresponding weekday | 10 |
| 9. | Implement the stack operations using python. | 11-13 |
| 10. | Implement the Queue operations using python. | 14-15 |
| 11. | Given 2 string s1,s2 return a first, middle and last character of each string. | 16-17 |
| 12. | Implement Shift key Cipher using python | 18-19 |
| 13. | Write a python program to implement Frequency distribution table for a file or a string | 20-21 |
| 14. | Write a Python function that takes a list of words and returns the longest word and the length of the longest one. | 22 |
| 15. | Write a python program to sort a tuple of tuples by 2nd item tuple1 = (('a', 23),('b', 37),('c', 11), ('d',29))  Expected output: (('c', 11), ('a', 23),('d', 29), ('b', 37)) | 23 |
| 16. | Implement Fermat's Test using Square and Multiply algorithm to get the probability of the number being Prime | 24 |
| 17. | Write a python program to get the probability of a number being prime implementing Rabin Muller test. | 25-27 |
| 18. |  | 28-29 |

**Experiment - 01**

Write a python program to print the below pattern.

a) 1

2 2

3 3 3

4 4 4 4

**Program Code:**

n=4

for i in range(1,n+1):

for j in range(i):

print(i,end=" ")

print()

**Output:**

1

2 2

3 3 3

4 4 4 4

b) 1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

**Program Code:**

n=6

for i in range(1,n):

for j in range(1,n):

print(j,end= " ")

print()

**Output:**

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

C) \* \* \* \*

\* \* \*

\* \*

\*

**Program Code:**

n=4

for i in range (n):

for j in range(n-i):

print("\*",end=" ")

print()

**Output:**

\* \* \* \*

\* \* \*

\* \*

\*

d) 1

3 2

6 5 4

10 9  8  7

**Program Code:**

start=1

stop=2

current\_num=stop

for row in range(2,6):

for col in range(start,stop):

current\_num-=1

print(current\_num,end=" ")

print("")

start=stop

stop+=row

current\_num=stop

**Output:**

1

3 2

6 5 4

10 9 8 7

**Experiment - 02**

Write a python function to find GCD of 2 numbers.

**Program Code:**

def compute(x, y):

while(y):

x, y = y, x % y

return x

a = int(input("Enter the value for a: "))

b = int(input("Enter the value for b: "))

# prints 12

print ("The gcd of ",a,"and",b," is : ",compute(a,b))

**Output:**

Enter the value for a:25

Enter the value for b: 30

The gcdof 25 and 30 is : 5

**Experiment - 03**

Write a python program to check whether the given string/number is palindrome or not?.

**Program Code:**

A='malayalam'

B=A[::-1]

if A==B:

print('The given string is Palindrome')

else:

print('It is not a Palindrome

**Output:**

The given string is Palindrome

**Experiment - 04**

Write a python program to print factorial of number.

**Program Code:**

n=int(input("Enter a num"))

fact = 1

if n == 0:

print('No fact')

elif n == 1:

print("Fact is 1")

else:

for i in range(1,n+1):

fact = fact\*i

print('Fact is',fact)

**Output:**

Enter a num: 5

Fact is 120

**Experiment - 05**

Write a python function to check whether the number is even or odd.

**Program Code:**

def check\_even\_odd(num):

if num % 2 == 0:

print(num, "is even")

else:

print(num, "is odd")

num= int(input("Enter a num"))

print(check\_even\_odd(num))

**Output:**

Enter a num: 6

6 is even

**Experiment - 06**

A school has following rules for grading system.

a)Below 25 is F

b)25 To 45 is E

c)45 to 50 is D

d)50 to 60 is C

e)60 to 80 is B

f)above 80 is A

Ask user to enter marks and print the corresponding grade.

**Program Code:**

marks = int(input("Enter your marks: "))

if marks < 25:

grade = "F"

elif marks >=25 and marks < 45:

grade = "E"

elif marks >= 45 and marks < 50:

grade = "D"

elif marks >= 50 marks < 60:

grade = "C"

elif marks >= 60 and marks < 80:

grade = "B"

else:

grade = "A"

print("Your grade is:", grade)

**Output:**

Enter your marks: 85

Your grade is: A

**Experiment - 07**

Write a program to produce the Fibonacci series

**Program Code:**

def fibonacci(n):

sequence = [0, 1]

for i in range(2, n):

next\_term = sequence[i-1] + sequence[i-2]

sequence.append(next\_term)

return sequence

print(fibonacci(10))

**Output:**

[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]

**Experiment - 08**

Write a python program to accept number from 1 to 7 and display the corresponding weekday.

**Program Code:**

day\_number = int(input("Enter a number between 1 and 7: "))

if day\_number == 1:

print("Monday")

elif day\_number == 2:

print("Tuesday")

elif day\_number == 3:

print("Wednesday")

elif day\_number == 4:

print("Thursday")

elif day\_number == 5:

print("Friday")

elif day\_number == 6:

print("Saturday")

elif day\_number == 7:

print("Sunday")

else:

print("Invalid input! Please enter a number between 1 and 7.")

**Output:**

Enter a number between 1 and 7: 5

Friday

**Experiment -09**

Implement the stack operations using python.

**Program Code:**

##############################

def push(stack,item,sp):

if(sp == size-1 ):

print("Stack overflow");

return(stack,sp)

sp = sp+1

stack[sp] = item

return(stack,sp)

##############################

def pop(stack,sp):

if(sp == -1 ):

print("stack is empty")

return(stack,sp,"null")

pop\_item = stack[sp]

stack[sp] = "null"

sp = sp - 1

return(stack,sp,pop\_item)

##############################

size= 3

sp = -1

stack = size\*["null"]

print(stack)

[stack,sp] = push(stack,'a',sp)

print(stack,sp)

[stack,sp] = push(stack,'b',sp)

print(stack,sp)

[stack,sp] = push(stack,'c',sp)

print(stack,sp)

[stack,sp] = push(stack,'d',sp)

print(stack,sp)

[stack,sp,item]=pop(stack,sp)

print(stack,item)

[stack,sp,item]=pop(stack,sp)

print(stack,item)

[stack,sp,item]=pop(stack,sp)

print(stack,item)

[stack,sp,item]=pop(stack,sp)

print(stack,item)

[stack,sp,item]=pop(stack,sp)

print(stack,item)

**Output:**

['null', 'null', 'null']

['a', 'null', 'null'] 0

['a', 'b', 'null'] 1

['a', 'b', 'c'] 2

**Stack overflow**

['a', 'b', 'c'] 2

['a', 'b', 'null'] c

['a', 'null', 'null'] b

['null', 'null', 'null'] a

**stack is empty**

['null', 'null', 'null'] null

**stack is empty**

['null', 'null', 'null'] null

**Experiment - 10**

Implement the Queue operations using python.

**Program Code:**

**#Enqueue Function**

def enqueue(queue,item,head,tail):

if (head == (tail + 1) or (head == 0 and tail == SIZE -1)):

print("\n Overflow cannot enqueue")

return(queue,item,head,tail)

if (tail >= 0) and tail <= (SIZE-1):

queue[tail] = item

tail = tail + 1

if tail == SIZE:

tail = 0

return(queue,item,head,tail)

**#Dequeue Function**

def dequeue(queue,head,tail):

if head==tail:

print("Queue is empty")

return(queue,"null",head,tail)

item=queue[head]

queue[head]="Null"

head=head+1

if head==SIZE:

head=0

return(queue,item,head,tail)

**#creation of empty queue of SIZE**

SIZE = 5

head = 0

tail = 0

queue = SIZE\*['Null']

print(queue,head,tail)

**#Enqueue**

[queue,item,head,tail] = enqueue(queue,10,head,tail)

print(queue,head,tail)

**#Dequeue**

[queue,item,head,tail]=dequeue(queue,head,tail)

print(queue,head,tail,item)

**Output:**

['Null', 'Null', 'Null', 'Null', 'Null'] 0 0

[10, 'Null', 'Null', 'Null', 'Null'] 0 1

['Null', 'Null', 'Null', 'Null', 'Null'] 1 1 10

**Experiment - 11**

Given 2 string s1,s2 return a first, middle and last character of each string.

**Program Code:**

s1 = input("Enter the first string1: ")

s2 = input("Enter the second string2: ")

**# getting the first, middle, and last characters of s1**

first\_char\_s1 = s1[0]

middle\_char\_s1 = s1[len(s1)//2]

last\_char\_s1 = s1[-1]

**# getting the first, middle, and last characters of s2**

first\_char\_s2 = s2[0]

middle\_char\_s2 = s2[len(s2)//2]

last\_char\_s2 = s2[-1]

**# printing the results**

print("First, middle, and last characters of the first string:")

print("First character:", first\_char\_s1)

print("Middle character:", middle\_char\_s1)

print("Last character:", last\_char\_s1)

print("\nFirst, middle, and last characters of the second string:")

print("First character:", first\_char\_s2)

print("Middle character:", middle\_char\_s2)

print("Last character:", last\_char\_s2)

**Output:**

Enter the first string1:"Happiness"

Enter the first string2:"Goal"

First, middle, and last characters of the first string:

First character: H

Middle character: i

Last character: s

First, middle, and last characters of the second string:

First character: G

Middle character: a

Last character: l

**Experiment - 12**

Implement Shift key Cipher using python.

**Program Code:**

**#For no value of K in Key Cipher**

s='icwagnfcp'

for k in range(0,26):

c = ' '

for m in s:

c = c + chr(((ord(m)-97)+k)%26+97)

print(c,k)

**Output:**

icwagnfcp 0

jdxbhogdq 1

keycipher 2

lfzdjqifs 3

mgaekrjgt 4

nhbflskhu 5

oicgmtliv 6

pjdhnumjw 7

qkeiovnkx 8

rlfjpwoly 9

smgkqxpmz 10

tnhlryqna 11

uoimszrob 12

vpjntaspc 13

wqkoubtqd 14

xrlpvcure 15

ysmqwdvsf 16

ztnrxewtg 17

auosyfxuh 18

bvptzgyvi 19

cwquahzwj 20

dxrvbiaxk 21

eyswcjbyl 22

fztxdkczm 23

gauyeldan 24

hbvzfmebo 25

##### **or**#####

s='keycipher'

k=2

c=''

d=''

**#Encrypting**

for i in s:

c = c + chr(((ord(i)-97)-k)%26+97)

print('Encrypted Form =',c)

**#Decrypting**

for n in c:

d = d + chr(((ord(n)-97)+k)%26+97)

print('Decrypted Form =',d)

**Output:**

Encrypted Form = icwagnfcp

Decrypted Form = keycipher

**Experiment - 13**

Write a python program to implement Frequency distribution table for a file or a string.

**Program Code:**

**#define function**

def len\_space(text):

count=0

for s in text:

if (s!=" "):

count=count+1

return(count)

###############################

**#define function**

def prepare\_table(text,freq,length):

for s in text:

if(s!=" "):

i=ord(s)-65

freq[i]=freq[i]+1

for i,f in enumerate(freq):

freq[i]=f/length

return freq

**#calling function**

text="When you have a dream,remember cryptography???"

freq=26\*[0]

length=len\_space(text)

ft=(prepare\_table(text.upper(),freq,length))

for f in ft:

print("%2.2f" %f)

**Output:**

0.10

0.02

0.02

0.02

0.15

0.02

0.02

0.07

0.00

0.00

0.00

0.00

0.07

0.02

0.05

0.05

0.00

0.12

0.00

0.02

0.02

0.02

0.02

0.00

0.15

0.00

**Experiment - 14**

Write a Python function that takes a list of words and returns the longest word and the length of the longest one.

**Program Code:**

def longest(words):

longest=" "

for word in words:

if len(word) >len(longest):

longest = word

return longest,len(longest)

words=['apple','banana','cherry']

print(longest(words))

**Output:**

('banana', 6)

**Experiment – 15**

Write a python program to sort a tuple of tuples by 2nd item  
tuple1 = (('a', 23),('b', 37),('c', 11), ('d',29))

Expected output: (('c', 11), ('a', 23),('d', 29), ('b', 37))

**Program Code:**

tuple1 = (('a', 23),('b', 37),('c', 11), ('d',29))

sort = sorted(tuple1,key = lambda x:x[-1])

print(sort)

**Output:**

[('c', 11), ('a', 23), ('d', 29), ('b', 37)]

**Experiment – 16**

Implement Fermat's Test using Square and Multiply algorithm to get the probability of the number being Prime

**Program Code:**

def sam(a,b,n):

c=1

while b!=0:

if b%2==1:

c=(c\*a)%n

a=(a\*a)%n

b=b//2

return c

###################

import random

n=17

b=10

k=10

ar=random.sample(range(2,n-1),k)

for a in ar:

c=sam(a,b,n) #Calling function

if(c!=1):

print("Composite Number")

if c==1:

print('Probability of being Composite Number is',(1/(2\*\*k))\*100)

print('Probability of being Prime Number is',(1-(1/(2\*\*k)))\*100)

**Output:**

Probability of being Composite Number is 0.09765625

Probability of being Prime Number is 99.90234375

**Experiment – 17**

Write a python program to get the probability of a number being prime implementing Rabin Muller test.

**Program Code:**

############################

def sam(a,n,N):

p=1

while n!=0:

if(n%2==1):

p = (p\*a)%N

n = n//2

a = (a\*a)%N

return p

################################

################################

N = 53;

N\_1 = N-1

s=0

while (N\_1%2 == 0):

N\_1 = N\_1//2

s = s+1

d = N\_1

print(s,d)

#print(sam(2,10,7))

k=0;

for a in range(2,N):

print(" a = ", a )

COND1 = 0; COND2 = 0; k = k+1

P = sam(a, d, N)

print("P=",P,"\n")

if( P == 1 or P == N-1 ):

COND1 = 1

for r in range(1,s):

P = sam(a, pow(2,r)\*d, N)

print("P=",P,"\n")

if( P == N-1 ):

COND2 = 1

break

if(COND1 == 1 or COND2 == 1):

print("The probability that N is prime is ", (1 - 1/pow(4,k))\*100.0 )

else:

print("Composite")

break

input()

**Output:**

2 13

a = 2

P= 30

P= 52

The probability that N is prime is 75.0

a = 3

P= 30

P= 52

The probability that N is prime is 93.75

a = 4

P= 52

P= 1

The probability that N is prime is 98.4375

a = 5

P= 23

P= 52

The probability that N is prime is 99.609375

a = 6

P= 52

P= 1

The probability that N is prime is 99.90234375

a = 7

P= 52

P= 1

The probability that N is prime is 99.9755859375

a = 8

P= 23

P= 52

The probability that N is prime is 99.993896484375

**Experiment – 18**

Write a python program to get the multiplicative inverse of the number

**Program Code:**

**Output:**