DAILY ASSESSMENT FORMAT

|  |  |  |  |
| --- | --- | --- | --- |
| Date | 27/05/2020 | Name: | Prajna |
| Course: | Digital signal processing | USN: | 4AL16EC047 |
| Topic: | FFT Using MATLAB, Study and Analysis of FIR and IIR, Filtering Signal,ECG Signal Analysis. | Semester &  Section: | 8 “A” |
| FORENOON SESSION DETAILS | | | |

****

Fast Fourier Transform Using MATLAB:

clear all;

close all;

clc;

fs=1000

ts=1/fs

dt=0:ts:5-ts

f1=10;

f2=30;

f3=70

y1=10\*sin(2\*pi\*f1\*dt)

y2=10\*sin(2\*pi\*f2\*dt)

y3=10\*sin(2\*pi\*f3\*dt)

y4=y1+y2+y3

subplot (4,1,1)

plot (dt, y1,'r')

subplot (4,1,2)

plot (dt, y2,'r')

subplot (4,1,3)

plot (dt, y3,'r')

subplot (4,1,4)

plot (dt, y4,'r')

nfft=length(y4)

nfft2=2^nextpow2(nfft)

ff=fft (y4, nfft2)

fff=ff (1: nfft2/2)

figure ()

xfft=fs\*(0: nfft2/2)/nfft2

plot(abs(fff))

ECG Signal Analysis Using MATLAB:

sig=load('ecg.csv')

plot(sig) xlabel('samples')

ylabel ('Electrical Activity')

title ('ECG Signal Sampled')

plot(sig,'ro')

beat\_count=0;

for k=2: length (sig)-1

if(sig(k)>sig(k-1) & sig(k)>sig(k+1) & sig(k)>1)

%k

%disp ('Prominent peak found');

beat\_count=beat\_count+1

end

end

beat\_count fs=100;

n=length(sig);

duration\_in\_sec=n/fs

duration\_in\_min=duration\_in\_sec/60;

BPM=beat\_count/duration\_in\_min

sig=sig (1:500)

hold off

figure ()

plot(sig)

|  |  |  |  |
| --- | --- | --- | --- |
| Date | 27/05/2020 | Name: | Prajna |
| Course: | PYTHON | USN: | 4AL16EC047 |
| Topic: | Key value pairing,Loops | Semester &  Section: | 8 “A” |
| AFTERNOON SESSION DETAILS | | | |

Looping through the key value pairs:

# create a mapping of state to abbreviation

states = {

'Oregon': 'OR',

'Florida': 'FL',

'California': 'CA',

'New York': 'NY',

'Michigan': 'MI'

}

# create a basic set of states and some cities in them

cities = {'CA': 'San Francisco',

'MI': 'Detroit',

'FL': 'Jacksonville'}

# add some more cities

cities['NY'] = 'New York'

cities['OR'] = 'Portland'

# print out some cities

print('-' \* 10)

print("NY State has: ", cities['NY'])

print("OR State has: ", cities['OR'])

# print some states

print('-' \* 10)

print("Michigan's abbreviation is: ", states["Michigan"])

print("Florida's abbreviation is: ", states['Florida'])

# do it by using the state then cities dict

print('-' \* 10)

print("Michigan has: ", cities[states['Michigan']])

print("Florida has: ", cities[states['Florida']])

# print every state abbreviation

print('-' \* 10)

for state, abbrev in states.items():

print(f"{state} is abbreviated {abbrev}")

# print every city in state

print('-' \* 10)

for abbrev, city in cities.items():

print(f"{abbrev} has the city {city}")

# now do both at the same time

print('-' \* 10)

for state, abbrev in states.items():

print(f"{state} state is abbreviated {abbrev}")

print(f"and has city {cities[abbrev]}")

print('-' \* 10)

# safely get an abbreviation by state that might not be there

state = states.get('Texas')

if not state:

print("Sorry, no Texas.")

# get a city with a default values

city = cities.get('FL', 'Does not Exist')

print(f"The city for the state 'TX' is: {city}")

Long code to reverse things:

* Value from key is simple just write a square bracket, or use get.
* But if you want to do the opposite which is find key for a value then you have write this code.

# run the below

print(list(cities.keys())[list(cities.values()).index('New York')])

#list from dict

list( cities.values()).index('New York')

#list of keys

list(cities.keys())

#looping through all

for key, value in states.items():

print(f"{key} key is abbreviated {value} value")

#get items

states.items()

city = cities.get('XXX', 'Does not Exist')

print(f"The city for the state is: {city}")

For loop:

the\_count = [1, 2, 3, 4, 5]

fruits = ['apples', 'oranges', 'pears', 'apricots']

change = [1, 'rupee', 2, 'paisa', 3, 'mudra']

# traverse through the list we created

for xxx in the\_count:

print(f"This is count {xxx}")

# traverse throught a string list we created

for fruit in fruits:

print(f"A fruit of type: {fruit}")

# also we can go through mixed lists too

for i in change:

print(f"I got {i}")

# we can also build lists, first start with an empty one

elements = []

# then use the range function to do 0 to 5 counts

for i in range(0, 6):

print(f"Adding {i} to the list.")

# append is a function that lists understand

elements.append(i)

# now we can print them out too

for i in elements:

print(f"Element was: {i}")

This one is for\_each\_xxx

elements = []

# then use the range function to do 0 to 5 counts

for xxx\_each in the\_count:

print(f"Adding {xxx\_each} to the list.")

# append is a function that lists understand

elements.append(xxx\_each)

Traverse and double the value

# this first kind of for-loop goes through a list

for eachvalue in the\_count:

print(f"This is count {eachvalue\*2}")

Other languages dont have "in" the one in "for xxx in listofelements" hence they have to do the it the length way!

for i in range(0, len(the\_count)):

print (the\_count[i],i)

element=[]

for i in range(6, 0, -1):

print(f"Adding {i} to the list.")

# append is a function that lists understand

elements.append(i)

List and changing things for printing.

the\_count = [10, 20, 30, 40, 50]

for xxx in the\_count:

print(xxx-1)

for i in range(0, 3):

print(i)

List with loop and append command.

# we can also build lists, first start with an empty one

elements = []

# then use the range function to do 0 to 5 counts

for p in range(0, 6):

print(f"Adding {p} to the list.")

# append is a function that lists understand

elements.append(p)

xxx =range(0, 6)

print (xxx)

list(xxx)

for i in range(4):

for j in range(4):

if j > i:

break

print((i, j))

It goes through list and double.

# this first kind of for-loop goes through a list

for eachvalue in the\_count:

print(f"This is count {eachvalue\*2}")

for i in range(0, len(the\_count)):

print (the\_count[i],i)

Loop in a loop:

for i in range(4):

for j in range(4):

# if j > i:

# break

print((i, j))

Or in a loop:

sum = 0

for i in range(10):

# % is the modulo operator

if i % 3 == 0 or i % 5 == 0:

sum =sum + i

print ("i",i)

print ("sum",sum)

Step size changed

for i in range(5,-5,-1):

if i >=-2:

print ('Non-negative')

else:

print('Negative')

Printing number triangle

def line(n):

triangle = ''

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

triangle = triangle + (str(i))

print(triangle)

i+=1

Pyramid world:

def half\_pyramid(rows):

print('Half pyramid...\n')

for i in range(rows):

print('\*' \* (i+1))

def full\_pyramid(rows):

print('\nFull pyramid...\n')

for i in range(rows):

print(' '\*(rows-i-1) + '\*'\*(2\*i+1))

def inverted\_pyramid(rows):

print('\nInverted pyramid...\n')

for i in reversed(range(rows)):

print(' '\*(rows-i-1) + '\*'\*(2\*i+1))

half\_pyramid(5)

full\_pyramid(5)

inverted\_pyramid(5)

Prime Number and nested loops:

def is\_prime(number):

# if number is equal to or less than 1, return False

if number <= 1:

return False

for x in range(2, number):

# if number is divisble by x, return False

if not number % x:

return False

return True

output = []

listofn = [262, 102, 23, 164, 96]

numtotest = 30

# this the main loop - it goes through each number

# once we get access to each then add things and also for prime number we create another function and check

def get\_divisors(listofn, numtotest):

for i in listofn:

print (i)

sumofn = 0

#This loop converts the elements like 262 into string and then sums them up

for num in list(str(i)):

sumofn = sumofn + int(num)

# this is the sum of the digits of the number

# if checking for both conditions

if (numtotest % sumofn ==0) and (is\_prime(sumofn) == True):

#adding to the empty list

output.append(i)

return output

While loop:

* While loops controls the iteration inside the loop.
* They are different from for loop where the iteration is fixed.

i = 0

numbers = []

while i < 7:

print(f"At the top i is {i}")

numbers.append(i)

i = i + 3

print("Numbers now: ", numbers)

print(f"At the bottom i is {i}")

print("The numbers: ")

for num in numbers:

print(num)

i = 0

numbers = []

while i < 6:

print(f"At the top i is {i}")

numbers.append(i)

i = i + 1

print("Numbers now: ", numbers)

print(f"At the bottom i is {i}")

print("The numbers: ")

for xxx in numbers:

print(xxx)

i=0

li = [2,3,5,67,85,5,4]

max = len(li)

while i < max:

print (li[i])

i=i+1

* Break in a while loop as while loop never ends.

x = 256

total = 0

while x > 0:

if total > 400:

break

total =total+ x

x = x / 2

While in a While:

line=1 #this is the initial variable

while line <= 5 :

pos = 1

while pos < line:

#This print will add space after printing the value

print (pos,)

#increment the value of pos by one

pos += 1

else:

#This print will add newline after printing the value

print (pos)

#increment the value of line by one

line += 1

* Another while inside a while,

i=1

while i < 11:

j = 0

while j < i:

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

j=j+1

print()

i=i+1

print("Rest of the program")