DAILY ASSESSMENT FORMAT

Date:	**2-06-2020**	**Name:**	**MOUNITHA D M**
Course:	**DIGITAL DESIGN USING HDL**	**USN:**	**4AL17EC055**
Topic:	**FPGA Basics:Architecture,Application and uses Verilog HDL Basics by Intel Verilog testbench code**	**Semester & Section:**	**6TH SEM "A" SEC**
Github Repository:	**Mounitha_-ec055**		

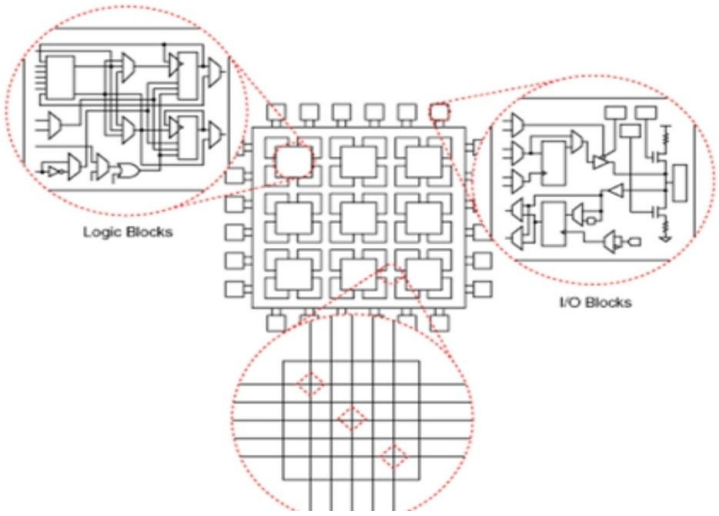
FORENOON SESSION DETAILS

Image of session

FPGA Architecture

A basic FPGA architecture (Figure 1) consists of thousands of fundamental elements called configurable logic blocks (CLBs) surrounded by a system of programmable interconnects, called a fabric, that routes signals between CLBs. Input/output (I/O) blocks interface between the FPGA and external devices.

Depending on the manufacturer, the CLB may also be referred to as a logic block (LB), a logic element (LE) or a logic cell (LC).



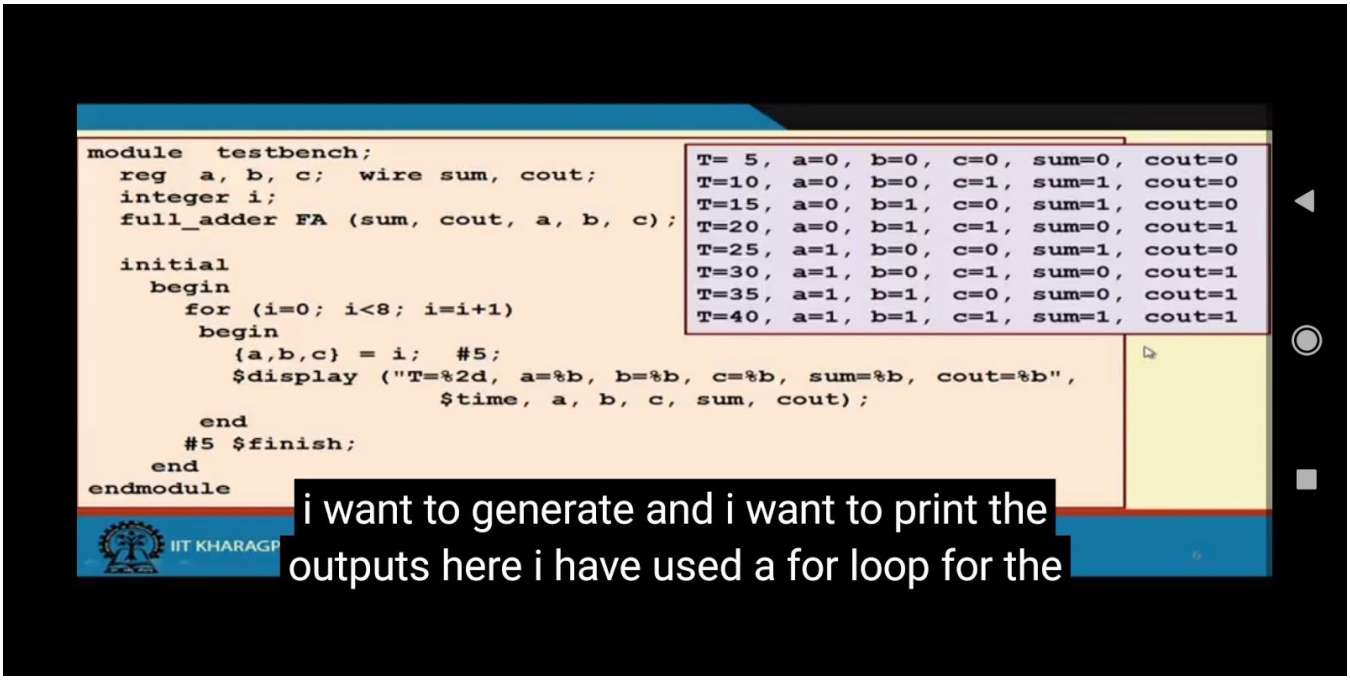
Arithmetic Operators

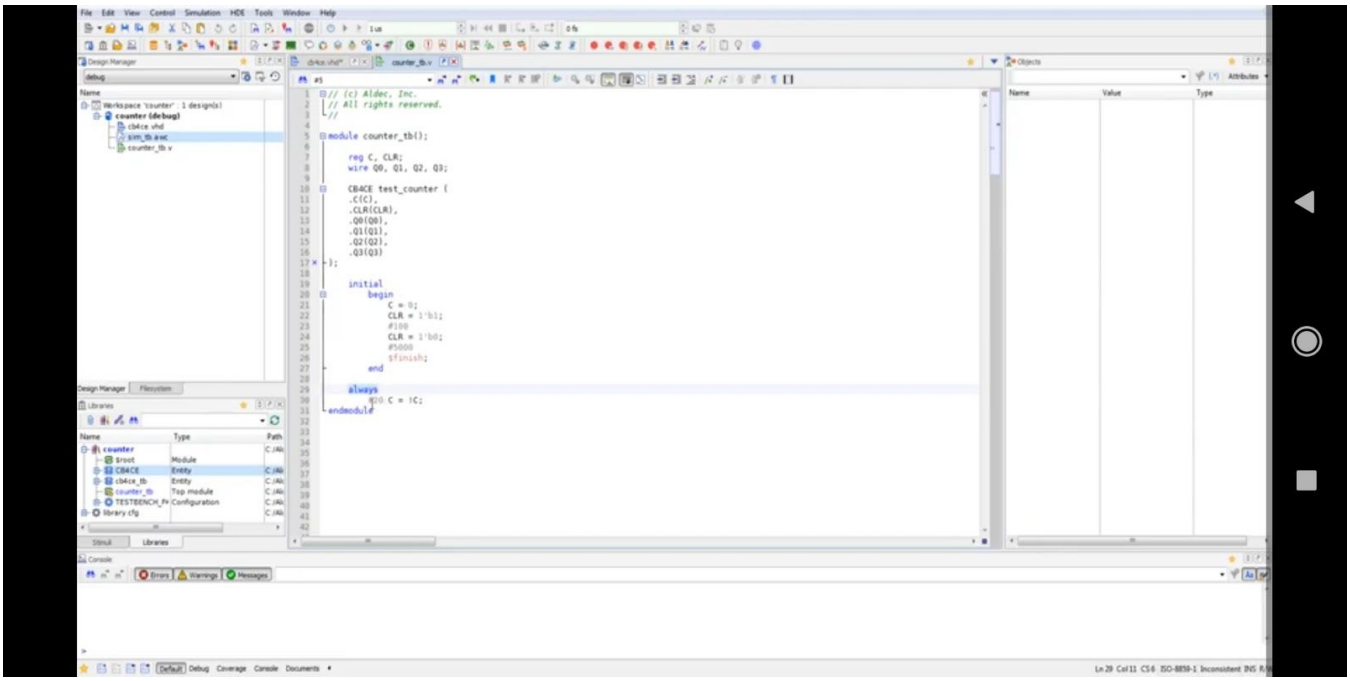
Operator Symbol	Functionality	Examples ain = 5 ; bin = 10 ; cin = 2'b01 ; din = 2'b0z		
+	Add, Positive	bin + cin ⇒ 11	+bin ⇒ 10	ain + din ⇒ x
–	Subtract, Negate	bin – cin ⇒ 9	-bin ⇒ -10	ain – din ⇒ x
*	Multiply	ain * bin ⇒ 50		
/	Divide	bin / ain ⇒ 2		
%	Modulus	bin % ain ⇒ 0		
**	Exponent*	ain ** 2 ⇒ 25		

- Treats vectors as a whole value
- Results unknown if any operand is Z or X
- Carry bit(s) handled automatically if result wider than operands
- Carry bit lost if operands and results are same size

** Check synthesis tool for support*

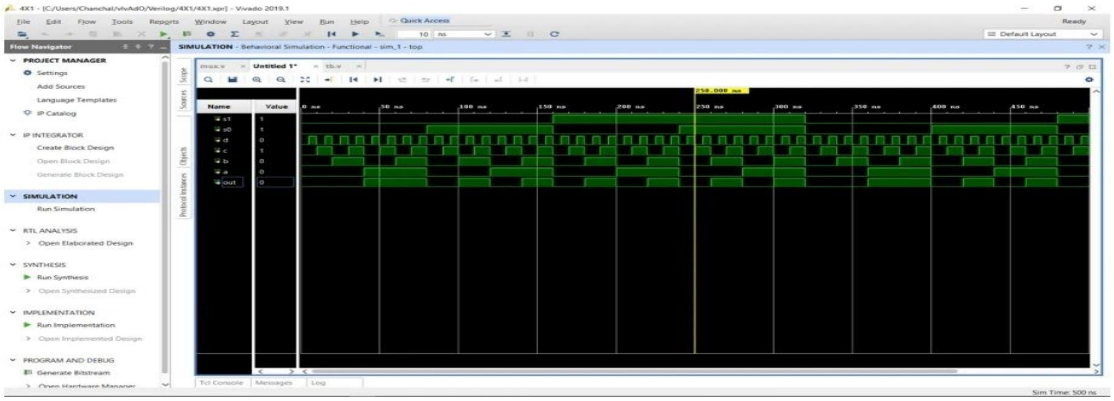
ALTERA. MEASURABLE ADVANTAGE™





Simulation Waveforms

The following window is the simulation log for the 4:1 multiplexer. The waveforms remain the same for all the styles of modeling.



Simulation Waveform 4×1 MUX

Report – Report can be typed or hand written for up to two pages.

Digital Design Using HDL

9/06/2020

Day2:

FPGA Basics Architecture Application and Uses.

→ The field programmable gate array (FPGA) is an integrated circuit that consists of internal hardware blocks with user programmable interconnects to customize operation for a specific application.

What is FPGA?

The field programmable gate array (FPGA) is an integrated circuit that consists of internal hardware blocks with user programmable interconnects to customize operation for a specific application.

FPGA architecture

A basic FPGA architecture (figure) consists of thousands of fundamental Elements called configurable logic blocks surrounded by a System of program

Verilog HDL Basics

What is Verilog

IEEE industry standard Hardware Description Language (HDL) use to disribe a digital system use in both hardware simulation & synthesis

Instantiation Format

<component_name> #<delay> <instance_name>

<component_name>

The module name of your lower level component

<delay>

delay through component

optional

<instance_name>

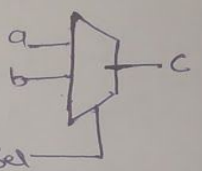
Blocking/Non Blocking Rule of Thumb

→ use blocking operator (=) for combinatorial logic
→ use nonblocking operator (<=) for sequential logic.

Combinatorial process

→ Sensitive to all inputs used in the combinatorial logic

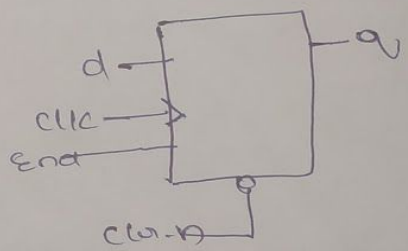
always @(a, b, sel) } sensitivity list includes all inputs used in the combinational logic.
always @*



Clocked process

- Sensitive to a clock /or and control signal

always @(posedge clk, negedge (rst_n)



Functions and Tasks

→ Uses
→ Replacing repetitive code
→ Enhancing readability

→ function
→ Return a value based on its inputs
→ produce combinational logic

→ Tasks
→ like procedure in other language
→ can be combinational or registered

```
module macc
output reg [15:0] out,
input [7:0] ina, inb,
input clk, rst_n,
End.
```

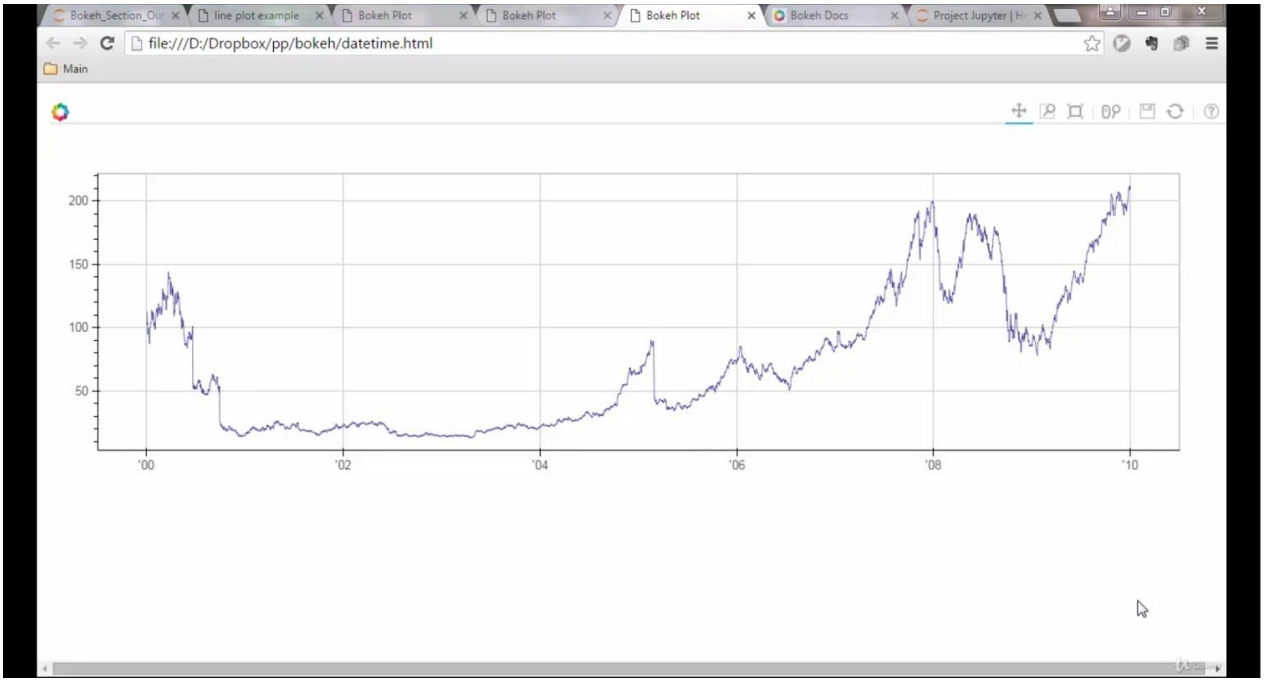
Task →
Implement a 4:1 MUX and write the test bench code to verify the module.

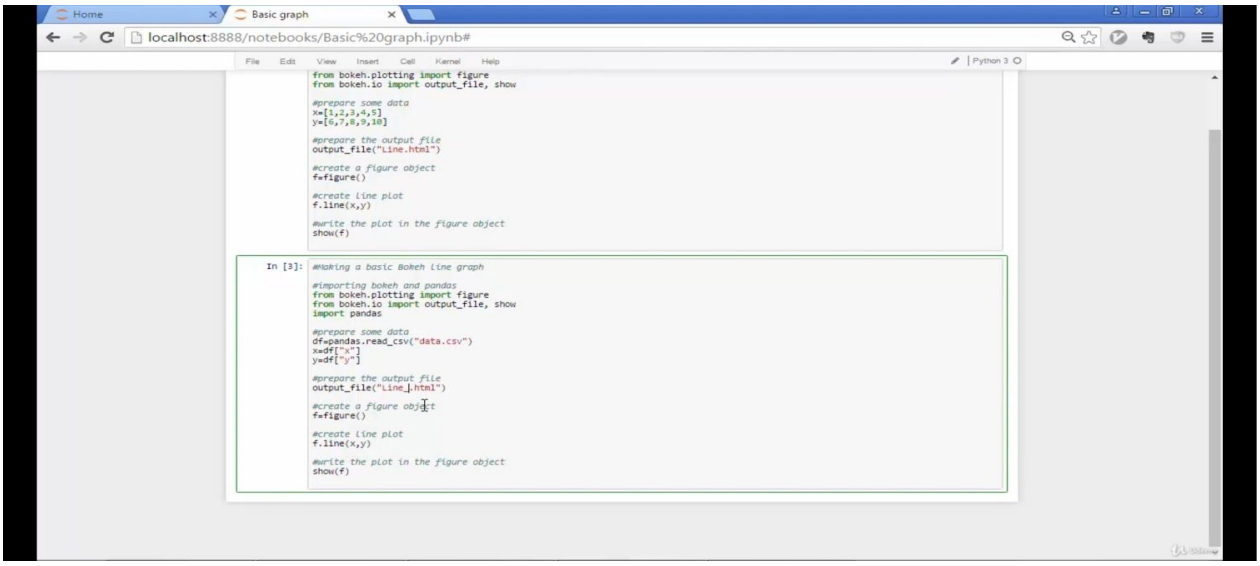
```
module top;
wire out;
reg a;
reg b;
reg c;
reg d;
reg S0, S1;
mux4 name (.out(out), .a(a), .b(b), .c(c)
initial
begin
a = 1'b0; b = 1'b0; c = 1'b0; d = 1'b0;
S0 = 1'b0; S1 = 1'b0;
#500 $finish
End
always #40 a = ~a;
always #20 b = ~b;
always #10 c = ~c;
always #5 d = ~d;
always #80 S0 = ~S0;
always #160 S1 = ~S1;
always @ (a or b or c or d or S0 or S1)
$monitor ("At time = %d, output = %d", $time, out
Endmodule
```

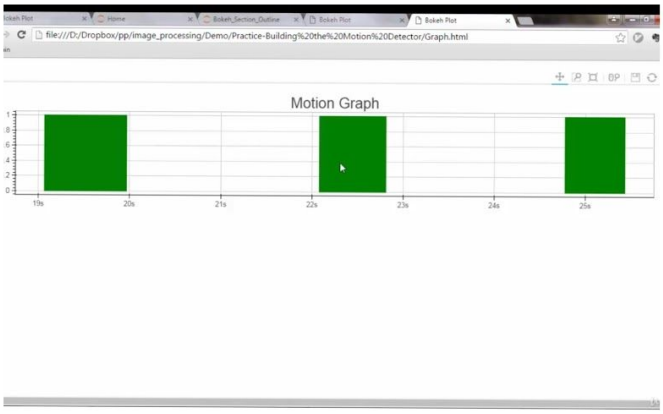
DATE	**2-06-2020**	**Name:**	**MOUNITHA DM**
Course:	**PYTHON**	**USN:**	**4AL17EC055**
Topic:	**Interactive Data visualization with Bokeh Webscraping with python Beautiful soup**	**Semester & Section:**	**6TH SEM "A" SEC**

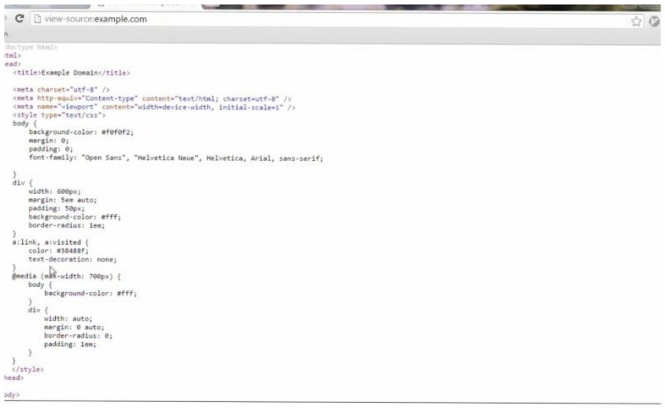
AFTERNOON SESSION DETAILS

Image of session









Lectures More

235 Plot Properties
Article

236 Plotting Weather Data (Practice)
Article

237 Solution
Article

238 Visual Attributes
Article

239 Time-series Plots
Video - 06:36 mins - Resources (2)

240 More Visualization Examples with...
Video - 04:21 mins

241 Plotting Time Intervals of the Motio...
Video - 14:05 mins - Resources (1)

242 Hover Tool Implementation
Video - 09:57 mins

Lectures More

242 Hover Tool Implementation
Video - 09:57 mins

Section 29 - Webscraping with Python Beautiful Soup

243 Section Introduction
Video - 01:57 mins

244 The Concept Behind Webscraping
Video - 04:30 mins

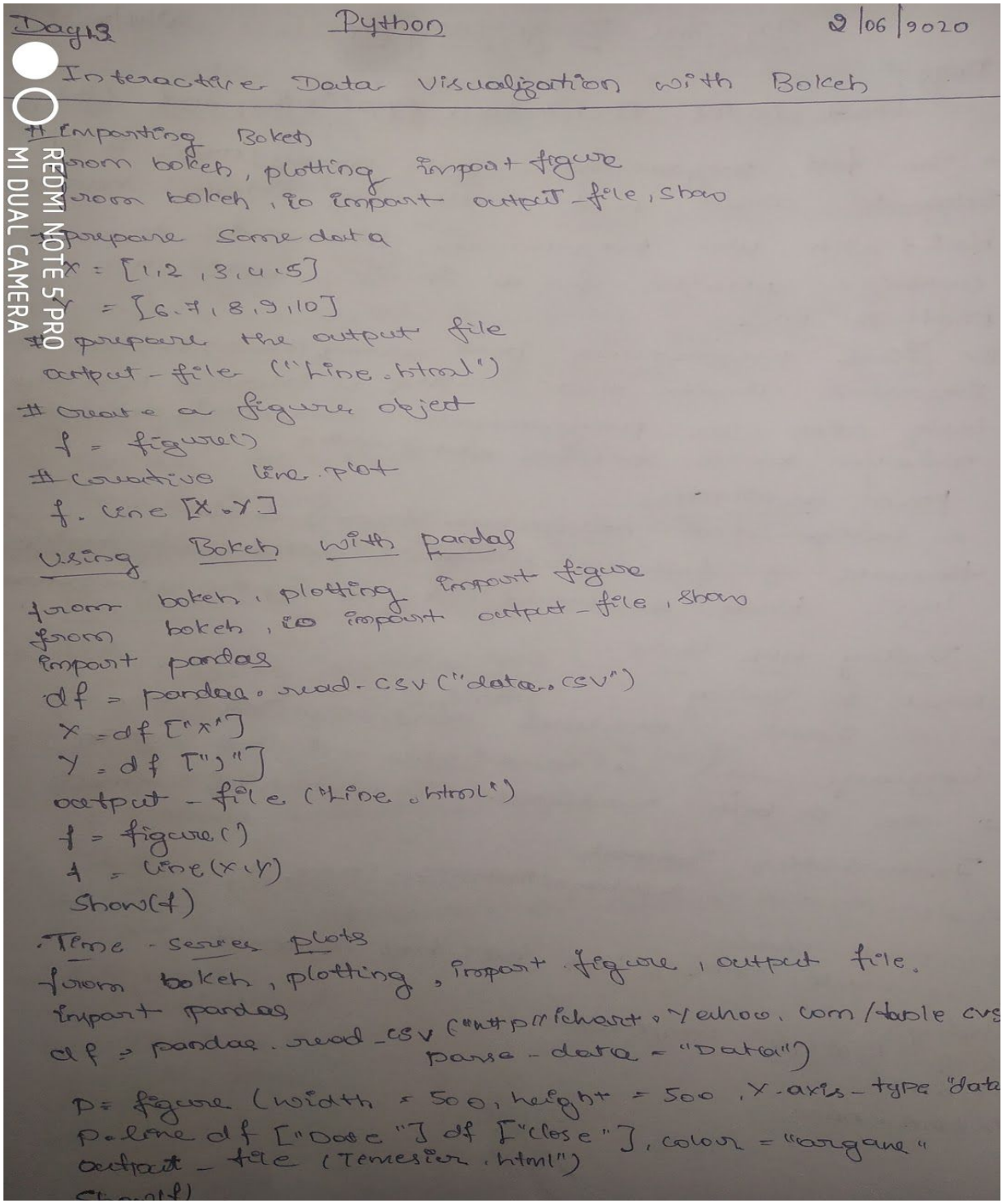
245 Request Headers
Article

246 Webscraping Example
Video - 16:22 mins

Section 30 - Application 7: Scrape Real Estate Property ...

247 Scraped Website Data - How The Ou...

Report – Report can be typed or hand written for up to two pages.



Day 13 — Python — 2/06/2020

Interactive Data Visualization with Bokeh

```
# importing Bokeh
from bokeh, plotting import figure
from bokeh, io import output_file, show
# prepare some data
x = [1,2,3,4,5]
y = [6,7,8,9,10]
# prepare the output file
output_file ("line.html")
# create a figure object
f = figure()
# creative line plot
f.line [x,y]
```

Using Bokeh with pandas

```
from bokeh, plotting import figure
from bokeh, io import output_file, show
import pandas
df = pandas.read_csv ("data.csv")
x = df ["x"]
y = df ["y"]
output_file ("line.html")
f = figure()
f = line(x,y)
show(f)
```

Time-Series Plots

```
from bokeh, plotting, import figure, output file.
import pandas
df = pandas.read_csv ("http://echart.../echoo.com/table cvs
                       parse_dates = "Date")
p = figure (width = 500, height = 500, x_axis_type "date
p.line df ["Date"] df ["close"], colour = "orange"
output_file ("Temesier.html")
show(f)
```