DAILY ASSESSMENT FORMAT

FORENOON SESSION DETAILS			
Date:	**25/6/2020**	**Name:**	**SAMRUDDI SHETTY**
Course:	**Python**	**USN:**	**4AL16IS047**
Topic:	**DAY 7**	**Semester & Section:**	**8th sem**
Github Repository:			

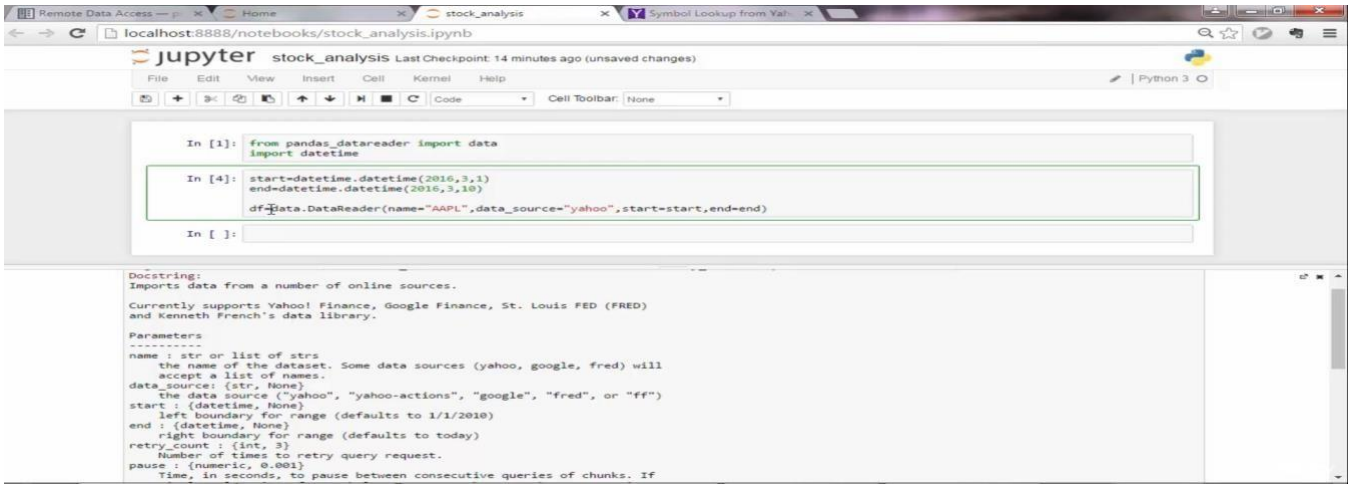
This section financial analysis with Python and interactive visualizations onthe web, so that's perfectly fine. And yeah to keep this short what you're going to learn in this section is you're going to learn how to get stock market data via Python, so without having to browse and download files from websites. You're going to get these data in Pandas data frames, so in tables inside Python and then they are ready to analyze.We're talking about the recent data,almost real-time and that's one thing and then once we get these data we'll analyze them using Python Pandas and we'll be working on a Jupiter notebook,and then we'll use the Bokeh library to visualize these data and without you should come up with this chart which you see in here. So it's a kind of picture which is used for visualizing stock market data. Yeah, this can be a bit complicated to build so you're going to learn a lot by creating this graph from scratch and once we create that graph, then we're going to embed that graph in in a website as you see in here.So this is the end program of this section and this is pure Python, so we're not going to use any JavaScript for this chart. So this is interactive, yeah.And you can even zoom in to certain areas if you like.

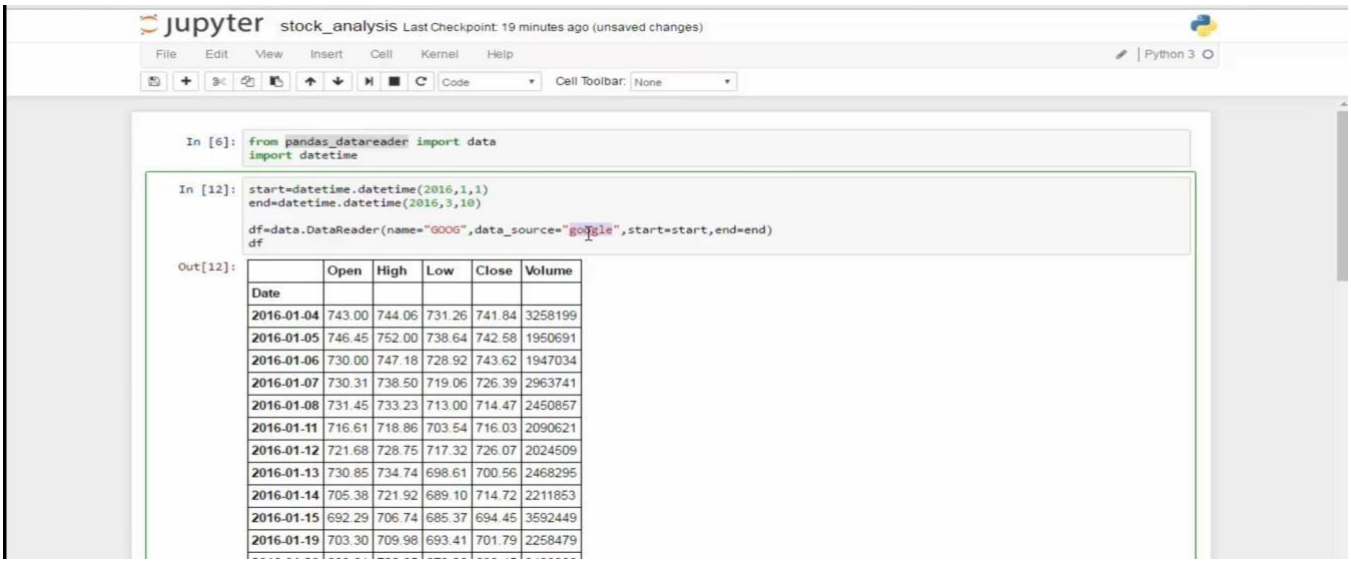
Program:

```
from flask import Flask, render_template

app=Flask(_name__)

@app.route('/plot/')
def plot():
    from pandas_datareader import data
    import datetime
    from bokeh.plotting import figure, show, output_file
    from bokeh.embed import components
    from bokeh.resources import CDN

    start=datetime.datetime(2015,11,1)
    end=datetime.datetime(2016,3,10)

    df=data.DataReader(name="GOOG",data_source="google",start=start,end=end)


    def inc_dec(c, o):
        if c > o:
            value="Increase"
```

```
        elif c < o:
            value="Decrease"
        else:
            value="Equal"
        return value

    df["Status"]=[inc_dec(c,o) for c, o in zip(df.Close,df.Open)]
    df["Middle"]=(df.Open+df.Close)/2
    df["Height"]=abs(df.Close-df.Open)

    p=figure(x_axis_type='datetime', width=1000, height=300,
sizing_mode="scale_both")
    p.title.text="Candlestick Chart"
    p.grid.grid_line_alpha=0.3

    hours_12=12*60*60*1000

    p.segment(df.index, df.High, df.index, df.Low, color="Black")

    p.rect(df.index[df.Status=="Increase"],df.Middle[df.Status=="Increase"],
           hours_12,
df.Height[df.Status=="Increase"],fill_color="#CCFFFF",line_color="black")

    p.rect(df.index[df.Status=="Decrease"],df.Middle[df.Status=="Decrease"],
           hours_12,
df.Height[df.Status=="Decrease"],fill_color="#FF3333",line_color="black")

    script1, div1 = components(p)
    cdn_js=CDN.js_files[0]
    cdn_css=CDN.css_files[0]
    return render_template("plot.html",
    script1=script1,
    div1=div1,
    cdn_css=cdn_css,
    cdn_js=cdn_js )



@app.route('/')
def home():
    return render_template("home.html")

@app.route('/about/')
def about():
    return render_template("about.html")

if_name_=="_main__":
```

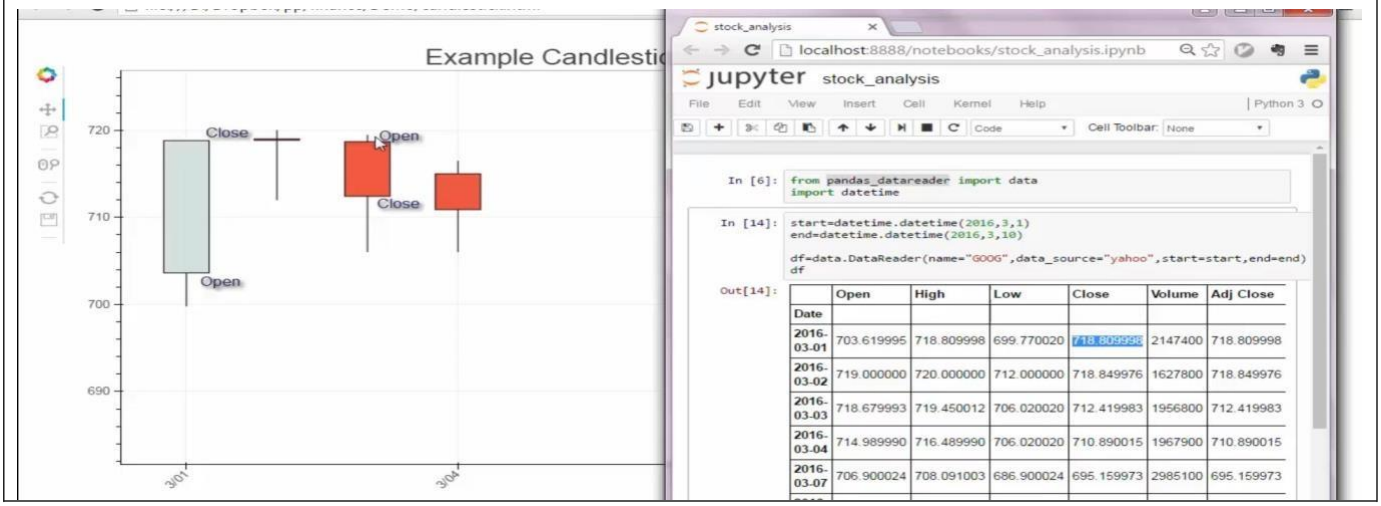
app.run(debug=True)**Downloading Various Datasets with Python:**





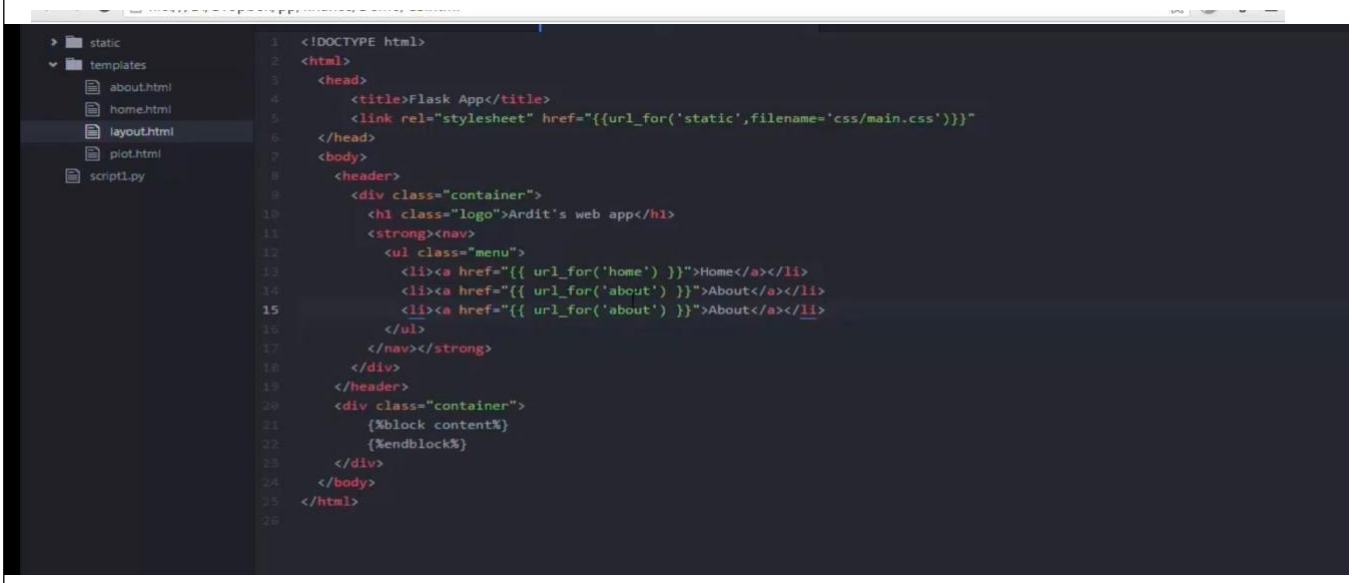
	Open	High	Low	Close	Volume
Date					
2016-01-04	743.00	744.06	731.26	741.84	3258199
2016-01-05	746.45	752.00	738.64	742.58	1950691
2016-01-06	730.00	747.18	728.92	743.62	1947034
2016-01-07	730.31	738.50	719.06	726.39	2963741
2016-01-08	731.45	733.23	713.00	714.47	2450857
2016-01-11	716.61	718.86	703.54	716.03	2090621
2016-01-12	721.68	728.75	717.32	726.07	2024509
2016-01-13	730.85	734.74	698.61	700.56	2468295
2016-01-14	705.38	721.92	689.10	714.72	2211853
2016-01-15	692.29	706.74	685.37	694.45	3592449
2016-01-19	703.30	709.98	693.41	701.79	2258479

Understanding Stock Market Data Candlestick Charts

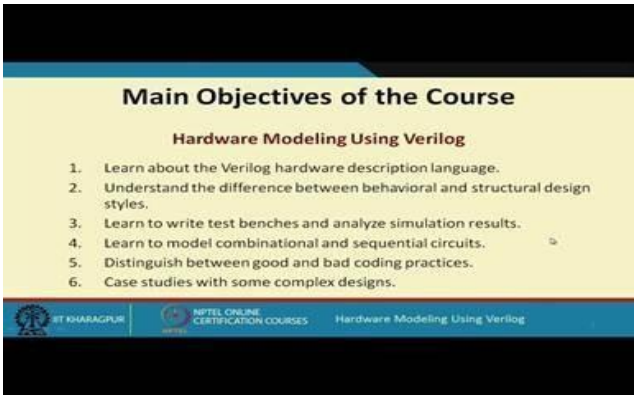


	Open	High	Low	Close	Volume	Adj Close
Date						
2016-03-01	703.619995	718.809998	699.770020	718.809998	2147400	718.809998
2016-03-02	719.000000	720.000000	712.000000	718.849976	1627800	718.849976
2016-03-03	718.679993	719.450012	706.020020	712.419983	1956800	712.419983
2016-03-04	714.989990	716.489990	706.020020	710.890015	1967900	710.890015
2016-03-07	706.900024	708.091003	686.900024	695.159973	2985100	695.159973

Building Candlestick Segments



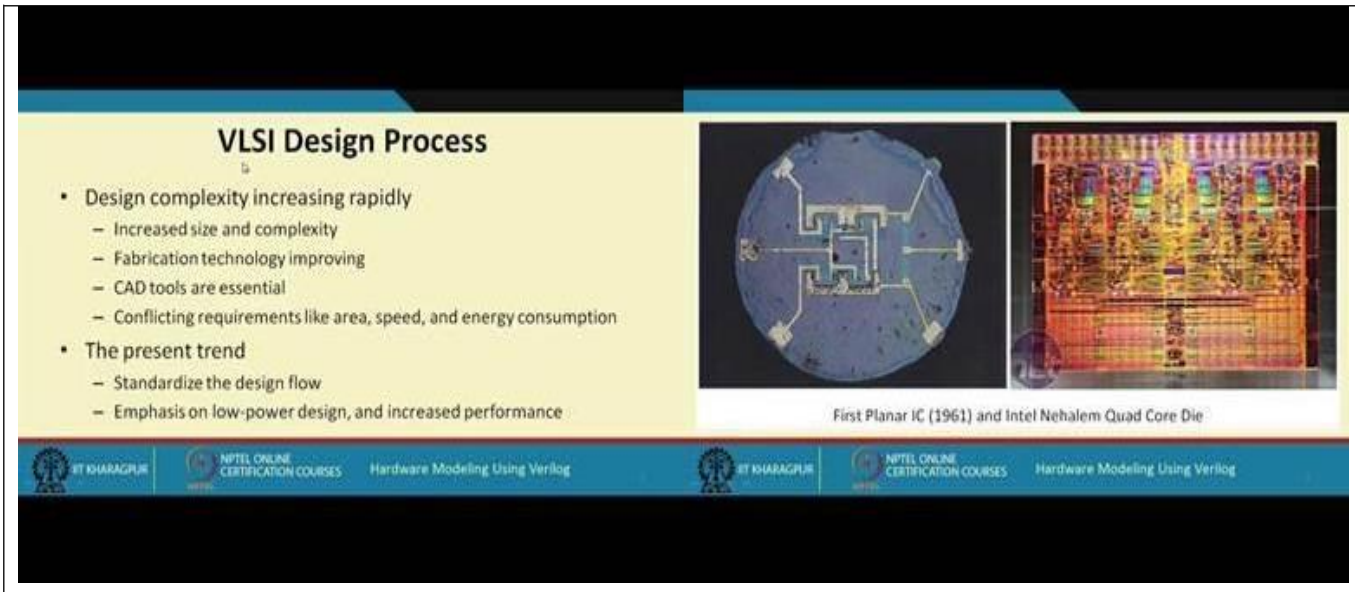
Date:	**04/6/2020**	**Name:**	**Ramanath Naik**
Course:	**HDL**	**USN:**	**4AL16EC054**
Topic:	**DAY 4**	**Semester & Section:**	**8th sem B sec**
Github Repository:			



So, let us start by talking about the main objectives of this course. So, as you know the name of this course is hardware modeling using Verilog. Now Verilog is one of the socalled hardware description languages that you may already be knowing, it is a language using which a designer can specify the behavior, or the functionality, or the structure of some given hardware; some specified hardware circuit. Now in this as part of this course well we shall of course, be learning about the Verilog hardware description language, its various features, the syntaxes, and so on.

Specifically, we shall be looking at, two different distinct way of modeling the functionality of a circuit this so-called behavioral and the structural design styles. So, we shall be explaining the differences. And from the point of view of verifying whether the design is correctly working or not, its correct or not, we have to write something called test benches, or test harness.

So, we shall also see how to write such test benches and evaluate the results of simulations, we shall be learning about modeling both combinational and sequential circuits. And during the course of this we shall be learning also what are the good practices and what are the so-called avoidable practices that a designer should be aware off. And of course, we shall be looking at some of the case studies. Specifically, at the end we shall be looking at the design of a complete processor and we see how using Verilog, we can design the data path and also the control path of the processor, ok.



So, when you talk about the VLSI design process there are a few things that we need to keep in mind. First thing is that over the years the complexity of VLSI circuits and consequently the design, they have increased dramatically. So, means when I say it is increase dramatically, means in fact there has been an exponential increase over the years. I shall show you a slide just depicting the kind of increase that has taken place. But the point to notice that earlier few decades back we use to design some chips, which consisted or contained few 100 or 1000 of gates or transistors, but now today we are talking about circuits or chips consisting of billions of transistors. So, you can just see the difference the dramatic advancements, and improvements that has been taken place over the years.

So, because of this increased size and complexity, this has been met possible of course because of improvement in the fabrication technology. The VLSI fabrication technology have improved dramatically. And as a consequence because of the complexity of the circuits manual design is simply ruled out. So, earlier when the circuit was smaller you could have designed your circuits on a piece of paper layout, on a piece of graph paper and so on and so forth, but now when we are talking about millions and billions of transistor you have to make use of a computer system and use some so-called computer aided design tools ok the CAD tools.

So, there are two computing HDLs today most popular: so one is Verilog other is VHDL. So, in this course as I have told you we shall be looking at the language Verilog. Now earlier as I have said that the designs are created using HDLs, so Verilog or VHDLs are very typical examples of these HDLs.

So, you can specify a design in either Verilog, or in VHDL and as the CAD tools transformed these designs from one level to the next, so the transform design is also expressed in similar kind ofhardware description languages.

Now, these are not the only ones there are other hardware description languages as well, some of the popular languages are like SystemC, SystemVerilog and so on, but here in this course we shall be concentrating only on Verilog. So, there are some other steps in the design flow also, these are not the only ones. Like simulation is very important step for verification. Like for example in this course we shall be extensively using simulation to check and verify the Verilog modules that we will be writing. Ok, we will also be informing you telling you how to do this simulation so that you can do or carry out the simulation yourself, right.

So, this simulation can be carried out at various levels of specification, logic level, transistor level, circuit level and so on. There is a step called formal verification, of course this will be beyond the scope of this course, where using some mathematical and formal techniques, you can check whether your designs are meeting the specifications or not. And again testability analysis test pattern generation is also very important. So, when you manufacture or designs some hardware, you will also have to test whether your final manufactured hardware is working correctly or not. Again this step is beyond the scope of this course we shall not be also talking about testability and testing here.So, with this we come to the end of this first lecture. In this lecture we have basically tried to give you an overview about what are the things we are expected to cover in this course and some basic concepts of VLSI design flow. Because understanding the VLSI design flow the process that is embedded there in, it will allow you to have a better understanding of how you can create a design using a so-called HDL, it can be either Verilog or VHDL as I said, so that the final hardware that will be generated in the process can be better in some sense. So, over the course of this lectures that will follow, we shall being trying to address these issues