Stage 1 – Testings

Stage 2 – Python Programming

Stage 3 – Automation – Robot Framework (Web automation/windows automation), Database

Stage 4 – Customized Framework

Stage 5 – CI/CD – Git & Jenkins

* Assessment for each stages.
  + Assessment – Feb 16
  + Assessment – Feb 21
  + Assessment – Feb 23/24
  + Assessment – Feb 28
  + Assessment – Mar 2
  + Assessment – Mar 4
* Final Caption Project

1. Software Testing
   1. It is the process of executing of programs with the intent of finding the errors as early as possible.
   2. Helps verifyting and validating the software as per the requirement
   3. Aims at
      1. Finding defects
      2. Demonstrating the lack of quality
      3. Gap in requirement and product
      4. Faith in the end product
2. Role played in the company

Trainee 🡪 System engineer 🡪 Senior System engineer 🡪 Technology Analyst/Business Analyst 🡪 Team leader/Technology lead/Test lead 🡪 Project Manager 🡪 Senior Project Manager 🡪 Delivery Manger

1. How product is developed in the market?
2. Software development model – Waterfall model, V-Model, Agile Scrum model
3. SDLC – Software Development Life Cycle
   1. Requirement Analysis
      1. Gathering & Analysis
      2. Documentation
      3. System input and System functions
   2. Design
      1. Design Specfication
      2. Technology
      3. Broken into small modules
      4. Funtional planning
   3. Development Phase
      1. Design Specfication is transformed as working model
      2. Develop component and intergrated
      3. Unit Testing – developers
   4. Testing
      1. Small component will be tested first & then major components
      2. Test Execution based on plan
      3. Report the defects to the team and fixes
      4. Testing – Sign off
   5. Maintenance
      1. Troubleshooting
      2. Support – help desk & training
      3. Latest updates software
4. Waterfall model

One should move to next phone after completing the preceding phase only

Advantages:-

* Simple
* Provides a structured way of doing things
* Models works when requirement is crystal clear.

Disadvantages:-

* Communication gap can result in disaster
* Poor resource allocation
* Need crystal clear requirement

1. V-Model
2. Agile Scrum model – iterative model

Vision is to develop application similar to teams for internal usage.

Product backlogs – login, signup, start meeting page, voice call, video call option, cancel meeting, add to calendar option, mail feature.

(assume we have – 100 product backlogs) – need to complete 1 year – 52 weeks

Scrum master – role is to understand the product backlogs, explain to everyone in the team, allocate backlogs to develop, controlling the sprint, sprint backlogs

100 product backlogs – 52 weeks

Total Sprints – 26 Sprints (considering each sprint as 2 weeks)

Sprint grooming – where review happens now a days. – before each sprint

Sprint 1 (8 sprint backlogs – decided before entering the sprint) – 2 weeks 🡪 some backlogs should be developed and tested (manual testing).

Sprint 2 (10 sprint backlogs) - 2 weeks 🡪 some backlogs should be developed and tested (progession testing) (manual testing), automation for sprint 1 will happen.

verifying the previous sprint feature is also working fine or not (regression testing for all sprint 1).

Sprint 3 (10 sprint backlogs) - 2 weeks 🡪 some backlogs should be developed and tested (progession testing) (manual testing), automation for sprint 2 will happen.

verifying the previous sprint feature is also working fine or not (regression testing for all sprint 2 & for sprint 1 check – automation will be triggered ).

....

Sprint 26 (4 Sprint backlogs) – 2 weeks 🡪 some backlogs should be developed and tested (progession testing) (manual testing), automation for sprint 25 will happen

verifying the previous sprint feature is also working fine or not (regression testing for 25th sprint manually and automation will be triggered for 24 sprint ).

1. Importance of testing:
   1. Loss of money & testing
   2. Customer Satification
   3. Stay in business
   4. Injury or leads to death
   5. Reliablity of the software
2. Necessity of testing
   1. Discover & prevent the defects
   2. Quality control assurance
   3. Gain customer confidence
   4. Early testing & defect removal is less expensive

Facts:

Ariane 5 explosion – (16 bit digits instead of 64 bit digits – june 4, 1996)

Microsoft’s anti-Unix site crashes – the sites were running open source unix and apache servers. So they tried to host their own server (IIS server)

1. Error – Failure – Defect

Error (mistake) – action perform by human (human mistakes)

Failure – inablity of the system to perform functionality according to its requirment.

Defect – mismatch of the actual and expected result.

Examples: adding two numbers

{

a= 50

b=20

c = a – b

print (c)

}

Output – 70 ( but output is 30)

1. Seven testing principle
   1. Testing shows presence of defects - Testing reduces proability of undiscovered defects remaining in the software
   2. Exhaustive testing is Impractical – Testing everything (all combinations of input and precondition ) is not feasible

OS – platfrom

Word

Excel

Gaming

* 1. Early testing – focused defined objective.
  2. Defect Clusting – small number of modules contains most of the defects discovered during pre-release or show most of the failure
  3. Pesticide paradox
  4. Testing Context – testing depends on the context. Act differenlty on different contexts
  5. Absence of error fallacy – If your software or system is unusable then it does not matter how many defects are found and fixed – it is still unusable.

1. Fundamental test process - Software Testing Life Cycle
   1. Test Planning
      1. Testing objective for the requirement
      2. Test Plan – Documentation
   2. Test Design
      1. Test case Designing
      2. Test Indentifying the test data
      3. Environment setup - Tools required
   3. Test Implementation
      1. Test Execution
      2. Test Result
      3. Log the bugs found
      4. Re-test
   4. Test Closure
      1. Environment Cleanup
      2. Process & reviewed in the meeting
      3. Lesson learnt – for upcoming cycles
      4. Archive
2. STLC Deliverables – Entry & Exit Criteria
3. Test Planning
   1. Requirement Traceablity Matrix (RTM)
   2. Test planning/Stratergy document
      1. Automation feasability
      2. Effort estimation
4. Test Design
   1. Test Scenario connected with proper requirement
   2. Test Case
   3. Test Data & Script
5. Test Execution
   1. Test result
   2. Bug report – Bug/Defect - priority & Severity
   3. Submitting the final executable report
6. Test Closure
   1. Test Summary Report
   2. Test Metrics
      1. Test coverage
      2. Defect priority & Severity
   3. Lesson learnt – for upcoming cycles
   4. Sign off to release the product

Requirement <-Scenario <-Test Cases

1. Responsibilty of QA engineer:
2. Analyse
3. Planning
4. Scenario
5. Test case
6. Test script & Data
7. Test execution – pass or fail
8. Bug/Defect tracking –
   1. Finding Bug/Defect & tracking it
   2. Retest after fixing
9. Status report
10. Testing levels
    1. Unit Testing
    2. Component Testing
    3. Integration Testing

Integration Testing Approach

* + 1. Big bang approch

Wait for the all part to get intergrate and then testing how well the parts are interacting.

When some module is not integrated then we use below concept to test

* + 1. Top down approch

Login ( create a dummy model)

Transfer

Withdrawal

FD

* + 1. Bottom up approch
       1. Login
       2. Transfer
       3. Withdrawal
       4. FD (create dummy model)

* 1. System Testing
     1. Concentrating the behavior of the entire system
     2. Functional and non-functional testing
  2. Acceptance Testing – end user/customer
     1. Operations Acceptance testing – performance, stress, volume.
     2. Compliance acceptance testing – safe at work, rules & regulation
     3. Alpha/Beta testing – refer slide

BDD/ATDD – majorly used for achieving the acceptance testing (popular way of acceptance testing)

1. Testing Types
   1. Functional Testing

Black box testing

* + 1. Requirement specific testing
    2. Business process based testing
  1. Non-Functional Testing
     1. Performance – reponse the application
     2. Stress – bug fixing at high load.
     3. load testing – loading the many user
  2. Structural Testing
     1. White box testing – can be done at development
  3. Testing for changes
     1. Confirmation Testing or retesting
        1. Making sure the defect are fixed or not
        2. If it is working fine, then we cannot conlude the sytem is fine.
     2. Regeression Testing
        1. Test carried out to check the modification done has not brought any new defects.
        2. To be executed whenever code changes.
        3. Suitable level for implemention of automation.
  4. Maintenance Testing
* Done on the application already in production and want to change the old system.
* Enhance the application to improve the response.
* Migration of application from one database to another database
* Migration of application from one platform to another
* All levels of testing like component, intergration, system, acceptance testing will happens here.

Test plan 🡪 anlysis and design 🡪 implementation 🡪 exit criteria

1. Test case design:- working on orange hrm application
   1. Login module / homepage
   2. Forgot your password module
2. Testing techniques
   1. Static testing – undergone without running the software code
      1. Walkthrough
      2. Inspection
      3. Review
   2. Dynamic testing – undergone with running software code
      1. Black box tesing
      2. White box testing
      3. Grey box testing
      4. Experience based testing
3. Review & Test Process:-

Informal review – no proper protocal to run it

Formal review:-

Roles

Moderator – leads the review.

Scribe – someone logs the defect raised in the meeting.

Author – unclear documentation / validating the defects.

Reviewers – review the material for defects. (your role)

Manager – make sure the review objective is met.

1. Black box testing/ Requirement specification based testing
   1. Equivalent partitioning testing

OTP – 4 digit

|invalid|valid |invalid

| <4 | =4 | >4

101 – invalid

1203 – valid

12345 – invalid

* 1. Boundary value analysis

OTP – 4 digit

Invalid|valid|invalid

999| 0000/1111 to 9999 | 10000

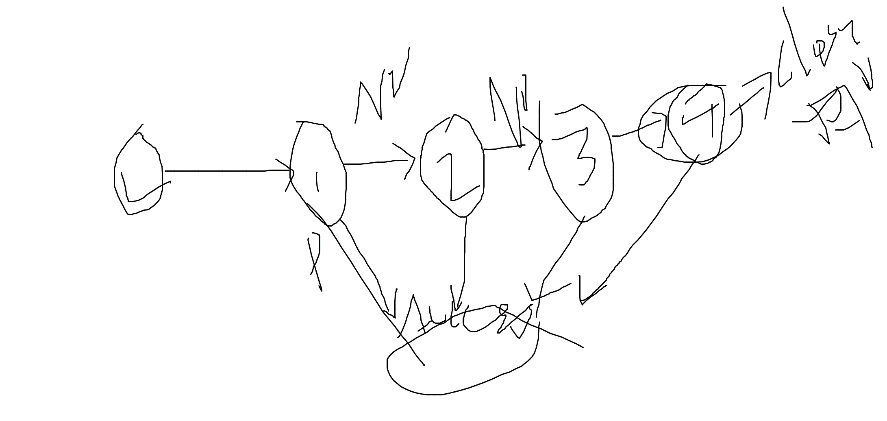
Testing the valid records – 0000/1111 and 9999

Invalid – 999 and 10000

* 1. Decision making

|  |  |  |
| --- | --- | --- |
| From | To | Search (enabled or disabled) |
| T | T | T |
|  |  |  |
| Any one for testing | | |
| F | T | F |
| T | F | F |
| F | F | F |

* 1. State transition / State chart



1. Expirence based testing - no format will be followed
   1. Exploratory testing – based on expirence testing will be conducted
   2. Ad-hoc testing – Situtation based testing
2. Structure based testing – by the developer. (we use some testing framework)
3. Testing Process
   1. Test planning and control
   2. Test Analysis and design
   3. Test Implementation and execution
   4. Test closure
4. Defect life cycle or bug life cycle
   1. Priority
      1. How soon the defect need to be fixed by developer?

P0 🡪 high priority

P1

P2

* 1. Severity
     1. Impact on the business

**Blocker** – Login page is crashed

**Critical** – Flight booking is not working

**Major** – Booking flight where user not able to confirmation message on screen but confirmation mail sent.

**Minor** – Spelling mistake

HS/HP - Login page is crashed

HS/LP – about us is crashed

LS/HP – logo mismatch/ spelling mistake

LS/LP – about us - spelling mistake/ logo not present

1. Screenshot in windows
   1. Press Windows+print screen key
   2. Use snipping tools to take screenshot
2. Test Management
   1. Test roles – lead & tester
   2. Test planning
   3. Estimation techinque

Accceptance Testing

* Use case model (points or diagram)
  + Informal format paragraph
  + One Paragraph doc
  + Formal paragraph

Required field for formal way of writing the use case:-

* + Use case name
  + Description
  + Pre-condition
  + Post condition
  + Basic steps
  + Alternative steps
  + Exception type
* User stories model (nowadays most of the project uses this model)

Add Employee Section – admin

Actor – Admin

Use case Name: Add Employee Section

Description: Admins are allowed to add an employee

Pre-Condition: Admin should login into orange hrm

Post-Condition – Admin should see employee in employee list section

Basic steps:

* Admin should login
* Admin should goto Add Employee Section
* Admin should be allowed to fill and add employee record

User stories model: (For gathering requirement (BDD) and also for acceptance testing (ATDD))

BDD – Behaviour Driven Development – concentrates on requirement.

ATDD – Acceptance Test Driven Development – concentrates on acceptance testing.

Scenarios:

* 1. Login
  2. Forgot password
  3. Add Employee

Scenario:

Given: Prequisite of the current scenario

When: Action/operation on the application

Then: validation/verification parts

BDD – requirement

Scenario: Valid Login

  Given: User should navigate to orangehrm url

  When: user provide valid credential

  Then: user should be redirected to user dashboard screen

ATDD

Gherkins language:-

Feature file (.feature)

Feature: Login

  In order to manage the employee portal

  As a user

  I would like access the portal

Scenario: Valid Login

  Given: User should navigate to orangehrm url

  When: user enter username as *'Admin'*

  And: user enter password as *'admin123'*

  And: user click on login

  Then: user should be redirected to user dashboard screen with content *'Assign Leave'*

Scenario: InValid Login

  Given: User should navigate to orangehrm url

  When: user enter username as *'Admin123'*

  And: user enter password as *'admin123'*

  And: user click on login

  Then: user should get error message ‘Invalid Credential’

Feature: Add Employee

    In order to add an employee

    As a admin

    I would like to access add employee section

Scenario: Add Employee with Enabled mode

  Given: Admin should login to the application

  When: Admin goto Add employee section

  And:  Admin entet firstname as ‘John’

  And: admin enter lastname as ‘J”

  Then: Admin should see the record in employee list

  And: Admin should see the same record in database

Scenario: Add Employee with Disabled mode

  Given: Admin should login to the application

  When: Admin goto Add employee section

  And:  Admin fill employee record

  | firstname | lastname | employee id |

  | John      | J        | 1245        |

  And: admin click on save

  Then: Admin should see the record in employee list

  And: Admin should see the same record in database

--------------------------------------------------------------------------------------------

Summary:

* + - 1. Definition of software testing
      2. Software development life cycle
      3. Software development model – waterfall, v-model, agile scrum model
      4. Importance/necessity of testing
      5. 7 testing principle
      6. Testing levels – Unit, Component, Intergration, System, Acceptancte testing.
      7. Testing types – functional testing/non-functional testing
      8. Testing process – planning & control, Analysis & Design, Implementation & Execution, Closure activity (summary report)
      9. Testing technique – Static & dynamic testing

JIRA – working with test case design and bug tracking.

X-Ray plugin – licenced plugin in jira for test management.

----------------------------------------------------------------------------------------------------------

UpperCamelCase – MyFirstProject

-------------------------------------------------------------------------------------------------------------

Git – Source Code Management – Open Source

Architecture of Git:

Local folder (local machine) 🡪 Local repository (in your local system) 🡪 Remote repository (github, bit bucket, AWS Code commit)

Installation:

Setup Git in your machine –

<https://github.com/git-for-windows/git/releases/download/v2.35.1.windows.2/Git-2.35.1.2-64-bit.exe>

Working with Git:-

Working on Local repo:-

1. Create a local repository – **git init**
2. Files planning to move 🡪 **git add .**
3. Update the local repository 🡪 **git commit** –m “day1 notes”

Working on Remote repo:

1. To register the github url using variable “origin”🡪 **git remote add origin** [**https://github.com/balaji-githubstore/NCSNotes.git**](https://github.com/balaji-githubstore/NCSNotes.git)
2. To update remote repo 🡪 **git push -u origin master**





Automation

Selenium – can automate only web application

Mobile/Windows automation – Appium

Programming language – Java, Python, C#, Javascript, Ruby, php, perl

Selenium – A Suite of tools - <https://www.selenium.dev/>

* + - 1. Selenium IDE
         1. No Programming knowledge required.
         2. Record and playback feature
         3. Plugin available – chrome, edge, firefox
         4. Use for simple scripting & exploratory testing
      2. Selenium RC – (Remote Control) – Depreciated
         1. Programming knowledge is must.
         2. Programming language – Java, Python, C#, Javascript, Ruby, php, perl
         3. Architecure

Source code (Python+Selenium RC) 🡪 RC Sever (Turn on/off) 🡪 Browser

* + - 1. Selenium WebDriver
         1. Programming knowledge is must.
         2. Programming language – Java, Python, C#, Javascript, Ruby, php, perl, Robot framework

Source code (***Robot framework+Selenium WebDriver****) 🡪 Browser*

* + - 1. Selenium Grid
         1. Run in multiple machine without actually moving the code
         2. Hub and Node

In our case, knowledge on basics of python and robot framework to work with selenium (web automation).

Python – Programming language

1. Architecture

Source code (.py - python code) 🡪 O/P

Python Interpreter

* Compiler (bytecode)
* Virtual machine

1. Installation
   1. Install python

Add Path to environment variable – PATH

C:\Program Files\Python310

C:\Program Files\Python310\Scripts

* 1. PyCharm IDE

<https://www.jetbrains.com/pycharm/download/#section=windows>

1. Create python project
2. Python Structure

Project (configure python interpreter)

Package

.py file

Methods/variables

UpperCamelCase – MyFirstProject

LowerCamelCase – myFirstProject

1. Datatypes (not a typed language)
   1. Numeric
      1. Integer
      2. Float
      3. Complex number
   2. Sequence types
      1. String
      2. List
      3. Tuple
   3. Boolean
   4. Dictionary – key/value pair

**Sequence type**

* 1. Zero based index

**List**

* 1. Declare the list
  2. Access the list item using index
  3. Get count of items in the list.
  4. Add the item in runtime

colors.append('blue')

* 1. Remove the item in runtime

colors.remove('yellow')

* 1. Add to particular index

colors.insert(1,'orange')

**List vs Tuple**

* Lists are mutable. Tuples are immutable
* For iteration, tuple will be fast. Tuple will be faster in response.
* Tuple consume less memory than list.

**Dictionary**

* 1. Key-value pair

1. Conditional statement
   1. If condition
   2. If else
   3. If elif
   4. If elif elif else
   5. Nested if condition
2. Relational operators

==, !=, >,<,>=,<=

1. Logical operator – AND, OR, NOT
2. Convert to string

num=25  
  
print("The result is " + str(num))

1. Convert to integer/float

a1="23"  
a2="24.9”  
print(type(a1))  
print(int(a1)+float(a2))

1. Iterative statements
   1. For loop
      1. Intialization
      2. End condition
      3. Incrementation
   2. Foreach – designed for handling collections
   3. While
2. Debugging the code

Add the Breakpoint

* 1. Terminate/stop
  2. Continue /resume
  3. Step into
  4. Step over

1. Methods/Functions – It’s a buliding block of the program
   1. Reusability

Instead of recreating logics in multiple place, we can create method in common file.

To create method

* Method\_name
* Parameters/arguments
* return

def method\_name(parameters):

res=45

return res

* Create and call the method inside same module.
* Create and call the method in different module.
  + Using import option
* Create and call the methods in class

1. String methods

<https://www.w3schools.com/python/python_strings_methods.asp>

1. Static vs non-Static (variable)

* Static variable/class variable
  + To access the static/class variable

Classname.variablename

* Non-static variable/instance variable
  + To access instance variable – use object ref

stu2 = student.Student(102, "Peter", 58.2)

print(stu2.student\_id)

* Static method
  + To access the static/class variable

Classname.methodname

* Non-static method
  + To access instance variable – use object ref

emp1=Employee(1001,"John",7889.2)

emp1.print\_employee\_detail()

1. Class and object

Class – a class is a template/blueprint/type (user defined datatype)

Object – is an instance of class

Object has state and behaviour

Notes:

1 GB – 1024 MB

1 MB – 1024 KB

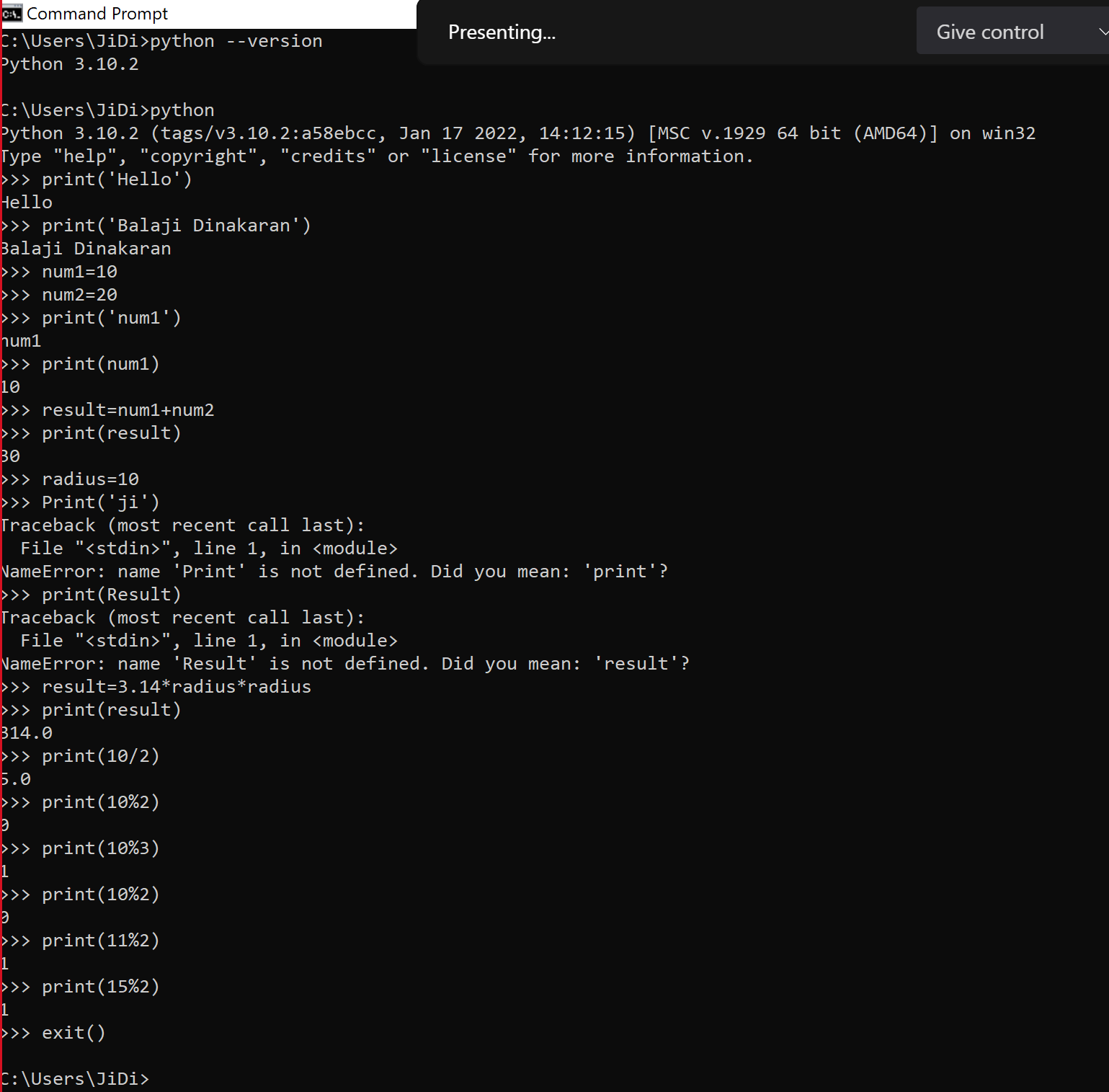
1 KB – 1024 B

1 B – 8 bits

1 bit – 0 or 1

//hello

Reference –



Session Task:-

Task 1

1. Create a list called fruits and store banana, mango, orange
2. Remove mango
3. Append pineapple
4. Insert record at 2nd index as JackFruit.
5. Print the fruits list

Task 2

1. Create a dictionary for keeping students record
2. To maintain one student record

studentId=1001

studentName=John

Percentage=72.2

[mailId=john@gmail.com](mailto:mailId=john@gmail.com)

Task 3

* + - 1. Declare the browsername and intialize with either ch or ff or edge or ie
      2. Write the if condition to launch the browser name based on the variable declared

1. Task 2

|  |  |
| --- | --- |
| **PERCENTAGE** | **GRADE** |
| 90 and above | A |
| 80 to 89 | B |
| 60 to 79 | C |
| 45 to 59 | D |
| below 45 | F |

Task 4

Print only numbers less than or equal to 50 from below list

numbers = [78, 66, 88, 22, 32, 8, 77, 7, 99, 9,11,30]

Task 5

Create a program to print the sequence

3 x 1 = 3

3 x 2 = 6

...

3 x 20 = 60

Task 5

*3.Write a program to remove 20 and print all the values using for each loop*list1 = [5, 10, 15, 20, 25, 50]

Task 6

1. Create package with name as “**math\_article**”
2. Create a python file (module) with name as “**Volume**”
3. Create method for below formula (any 3 methods)

|  |  |
| --- | --- |
| [Volume Of Sphere](https://byjus.com/maths/volume-of-sphere/) | [Volume Of A Cylinder](https://byjus.com/maths/volume-of-a-cylinder/) |
| [Volume Of A Pyramid](https://byjus.com/maths/volume-of-a-pyramid/) | [Volume Of Cone](https://byjus.com/maths/volume-of-cone/) |
| [Volume Of Cuboid](https://byjus.com/maths/volume-of-cuboid/) | [Volume Of Hemisphere](https://byjus.com/maths/volume-of-hemisphere/) |

Task 7

1. Create package with name as “**employee\_package**”
2. Create a python file (module) with name as “**employee\_module**”
3. Create class Employee template with employee id, employee name, employee salary, company name, company address.

emp1🡪 (emp\_id=1001,emp\_name=”John”,emp\_salary=89852,company\_name=NCS,company\_address=Pune)

emp1🡪 (emp\_id=1002,emp\_name=”Kenn”,emp\_salary=78888,company\_name=NCS,company\_address=Pune)

Task 8

* + - 1. <https://edabit.com/challenge/xbZR26rHMNo32yz35>
      2. <https://edabit.com/challenge/Yx2a9B57vXRuPevGh>
      3. <https://edabit.com/challenge/QQp2o22huzBCkHesy>
      4. <https://edabit.com/challenge/uKPc5faEzQkMwLYPP>
      5. <https://edabit.com/challenge/cXnkmRdxqJrwdsP4n>

Task 9

1. Create the package as “shopping”
2. Create a module as “shopping\_cart.py”
3. Create a class with name as “Item”
4. Declare variable as id , description , quantity , price.
5. Create a module as “shopping\_cart\_test.py”
6. Instantiate 3 object for item class and load below record

**Item 1 🡪 1, Shirt, 120, 900**

**Item 2 🡪 2, Laptop, 6, 45000**

**Item 3 🡪 3, Phone, 5, 12500**

1. Create a method to display item details
2. Create a method to calculate discounted item price by accepting discount percentage as parameter