**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **19-May-2020** | **Name:** | **Raziya Banu** |
| **Course:** | **TCS iON** | **USN:** | **4AL16EC058** |
| **Topic:** | **Resume, GD** | **Semester & Section:** | **8th sem & ‘B’ section** |
| **Github Repository:** |  |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
| **Report –**  In my first session today I have studied about the Resume making and the group discussion (GD).  **Resume:**  A resume is an important tool for your job search because it offers a page or two where you can display your top skills and qualities. However, a resume is much more than that. Resumes help employers make hiring decisions and help you get your first interview. That's why it matters how you structure your resume and what information you decide to include. In this article, you'll learn why a resume is important and get actionable resume tips that may help you achieve your next career move.  Resumes, usually accompanied by customized cover letters, get sent to employers to determine your eligibility and qualifications for a job. Employers use resumes to get a deeper understanding of candidate skills, strengths and experience. Your resume should reflect achievements, awards, education, experience and any other outstanding accomplishments that align with your career path and goals. Your resume is your first point of contact with the employer and sets the tone for subsequent steps such as first interview, second interview, pre-screening and on-boarding.  It can be difficult to discern which resume items are most helpful to include when applying for a job. A resume should feature your professional and personal strengths in a comprehensive yet concise manner. The Wisconsin Job Center has a helpful and thorough [guide to composing a resume](http://www.wisconsinjobcenter.org/publications/9433/9433.htm). If you want to avoid unnecessary or unhelpful information while featuring the most relevant and useful aspects of your experience and qualifications, then this brief guide should set you one the right track. Group Discussion: An average GD usually features 10 to 15 participants. The GD process begins by the announcement of the topic to the group, which is (usually) followed by a preparation time of 3 to 5 minutes. More than 5 minutes’ prep time may be given only if the GD is a case-study discussion, and has a long case statement.  At the end of the prep time, the panel signals the group to commence the discussion, and from then on plays the role of a non-participating observer. This means that the discussion is not moderated or ‘anchored’ by a panellist. The group members must discuss the topic as they deem appropriate without any kind of suggestion from the panel. The panel expects no particular order of speakers to be followed nor a minimum or maximum duration of speaking to be followed by individual participants.  The average duration of most GDs is 15 minutes (not including the prep time). In some exceptional cases (such as IIFT), the GD may continue for up to 45 minutes. One must remember that the longer the GD goes on, the more seriously the panel looks at the quality of the content (facts, analysis, explanation and argument) of the participant. The panel usually consists 3 or 4 panellists, who look at various aspects of the participants’ content and delivery. Please remember that the panellists may end the GD whenever they want to, and also extend the GD for as much as they want to. Nobody among the participants is supposed to keep time for the group or act on the assumption that the GD will end after the 15th minute. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date:** | **19-May-2020** | **Name:** | **Raziya Banu** | |
| **Course:** | **Udemy** | **USN:** | **4AL16EC058** | |
| **Topic:** | **Basics of Anaconda in Python** | **Semester & Section:** | **8th sem & ‘B’ section** | |
| **AFTERNOON SESSION DETAILS** | | | |
| **Image of session** | | | |
| **Report –**  In my session session I have studied about the basics of anaconda in Python and difference between the different versions of Python. **Getting started with Anaconda:** Anaconda Individual Edition contains [conda](https://conda.io/en/latest/) and [Anaconda Navigator](https://docs.anaconda.com/anaconda/navigator/), as well as Python and hundreds of scientific [packages](https://docs.anaconda.com/anaconda/packages/). When you installed Anaconda, you installed all these too.  Conda works on your command line interface such as Anaconda Prompt on Windows and terminal on macOS and Linux.  Navigator is a desktop graphical user interface that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands.  We can try both conda and Navigator to see which is right for you to manage your packages and environments. You can even switch between them, and the work you do with one can be viewed in the other.  Try this simple programming exercise, with [Navigator](https://docs.anaconda.com/anaconda/user-guide/getting-started/#nav-hello) and the [command line](https://docs.anaconda.com/anaconda/user-guide/getting-started/#cli-hello), to help you decide which approach is right for you.  **Python 3.0:**  Python 2 made code development process easier than earlier versions. It implemented technical details of Python Enhancement Proposal (PEP). Python 2.7 (last version in 2.x ) is no longer under development and in 2020 will be discontinued.  On December 2008, Python released version 3.0. This version was mainly released to fix problems which exist in Python 2. The nature of these change is such that Python 3 was incompatible with Python 2. It is **backward incompatible** Some features of Python 3 have been backported to Python 2.x versions to make the migration process easy in Python 3.  As a result, for any organization who was using Python 2.x version, migrating their project to 3.x needed lots of changes. These changes not only relate to projects and applications but also all the libraries that form part of the Python ecosystem.  prime reasons for using Python 3.x versions:   * Python 3 supports modern techniques like AI, machine learning, and data science * Python 3 is supported by a large Python developer's community. Getting support is easy. * Its easier to learn Python language compared to earlier versions. * Offers Powerful toolkit and libraries * Mixable with other languages   When it comes to Python version 2 vs. 3 today, Python 3 is the outright winner. That's because Python 2 won't be available after 2020. Mass Python 3 adoption is the clear direction of the future.  After considering declining support for Python 2 programming language and added benefits from upgrades to Python 3, it is always advisable for a new developer to select Python version 3. However, if a job demands Python 2 capabilities, that would be an only compelling reason to use this version. KEY DIFFERENCE:  * Python 3 syntax is simpler and easily understandable whereas Python 2 syntax is comparatively difficult to understand. * Python 3 default storing of strings is Unicode whereas Python 2 stores need to define Unicode string value with "u." * Python 3 value of variables never changes whereas in Python 2 value of the global variable will be changed while using it inside for-loop. * Python 3 exceptions should be enclosed in parenthesis while Python 2 exceptions should be enclosed in notations. * Python 3 rules of ordering comparisons are simplified whereas Python 2 rules of ordering comparison are complex. * Python 3 offers Range() function to perform iterations whereas, In Python 2, the xrange() is used for iterations. | | | |
|  | | | |