**EXECUTING THE GIVEN CODE**

**Step 1**: The given code is entered in an IDE like interface/ a “.dsl ” formatted file.

Sessions can be added a bit later (For multiple activities at the same time).

1. DSL Code is written by the user.
2. He now runs the code.
3. Error detection and correctness of the code is to be computed.

**Step 2**: The given code written by the user is read by the python file that is capturing data.

1. The Python code now reads the file.
2. DSL Code has to be split and stored in a list.
3. We have to make a clear distinction in segregating a keyword and a parameter. Following which,
4. All “key value” pairs are to be identified and stored in a dictionary separately.
5. Consider the command: “add int 1 2”. Here the key words are “add” and “int” and parameters are [1,2], which should yield 3. For this purpose, we write the add() function for the “add” in another python file called module.py that has all modules/operations.
6. Parameters are to be identified: We have to parse the entire code written by the user, break the code into different tokens with the help of lexers by identifying the different keywords provided by the user. Each token is divided by identifying the keyword until parsing to the next keyword.
7. Once we have divided the entire code into different tokens based on the keywords, we now are able to get a comprised format of keywords and parameters token-wise.
8. Hence, we create functions for these keywords in a module.py and link it to the main.py python file by using the “importlib” module.
9. We then execute the code on the command prompt EX: “Python main.py user\_code.dsl”.

**Step 3**: The code now needs to be **interpreted and checked for any errors**. These errors should include checking for **structural integrity**, **checking for syntax**, checking for **constraints/ parameters/ arguments**.

1. **Structural integrity**: Structural integrity involves the structure of the code written by the user. That is if the order of the code is in the correct sequence.
2. **Checking for Syntax**: Our DSL isn’t strongly typed. The syntax is fairly simple, hence the versatility of the users available using the language. We need to check for the required syntax.
3. **Check for constraints/ parameters**: Once we obtain the values of keyword (function) and the parameters, we need to check for the correctness such as number of parameters, order of the parameters, correctness of the parameter value entered by the user.

**Step 4**: **Executing** the **DSL code** written by the user after running the code on the command prompt:

1. Fetch the functions associated with the keywords entered in the DSL code.
2. Fetch the attributes associated with the respective functions.
3. Obtain the order of the occurrence of keywords as well as its parameters.
4. Execute the code line by line after error checking and the above steps following that.

**Step 5**: After executing the code, the first thing that would take place is for the information retrieval. For this, we first need to fetch the requested URL.

**Step 6**: After fetching the URL, we need to be granted the **access** to the URL being fetched. After the access is provided,

Using the commands from the DSL, we need to retrieve the required information from the URL. Refer to Step 1,2,3 and 4 for executing the code.

**Step 7**: After fetching the required contents, we need to **store** these **contents** in a file. The different types of retrievals are: 1) Text info 2) Files [Include: Multi-media, Documents and executable files] 3) Meta Data 4) Links.

**Step 8**: These retrieved files are to be stored in a folder/ Directory.

**Step 9**: End of operation.