**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **20 May 2020** | **Name:** | **Srinidhi J C** |
| **Course:** | **Python** | **USN:** | **4AL16EC078** |
| **Topic:** | **Loops and File processing** | **Semester & Section:** | **8th–Sem, B-Sec** |
| **Github Repository:** | **https://github.com/alvas-education-foundation/SrinidhiJC078.git** |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **FORENOON SESSION DETAILS** | | | |
|  | | | |
| **Report –**  In today’s section I have learnt these many programs:   * A Python program can get **user input** via the input function: * The **input** **function** halts the execution of the program and gets text input from the user**:**   name = input("Enter your name: ")   * The input function converts any **input to a string**, but you can convert it back to int or float:   experience\_months = input("Enter your experience in months: ")  experience\_years = int(experience\_months) / 12   * can **format strings** with (works both on Python 2 and 3):   name = "Sim"  experience\_years = 1.5  print("Hi %s, you have %s years of experience." % (name, experience\_years))  Output: Hi Sim, you have 1.5 years of experience.   * can also **format strings** with (Python 3 only):   name = "Sim"  experience\_years = 1.5  print("Hi {}, you have {} years of experience".format(name, experience\_years))  Output: Hi Sim, you have 1.5 years of experience.   * **For loops** are useful for executing a command over a large number of items. * You can create a **for loop** like so:   for letter in 'abc':  print(letter.upper())  Output:  A B C   * The name after for (e.g. letter) is just a variable name * can loop over **dictionary keys**:   phone\_numbers = {"John Smith":"+37682929928","Marry Simpons":"+423998200919"}  for value in phone\_numbers.keys():  print(value)  Output:  John Smith Marry Simpsons   * can loop over **dictionary values**:   phone\_numbers = {"John Smith":"+37682929928","Marry Simpons":"+423998200919"}  for value in phone\_numbers.values():  print(value)  Output:  +37682929928 +423998200919   * can loop over **dictionary items**: * phone\_numbers = {"John Smith":"+37682929928","Marry Simpons":"+423998200919"} * for key, value in phone\_numbers.items():   print(key, value)  Output:  ('John Smith', '+37682929928')  ('Marry Simpons', '+423998200919')   * **While loops** will run as long as a condition is true: * while datetime.datetime.now() < datetime.datetime(2090, 8, 20, 19, 30, 20):   print("It's not yet 19:30:20 of 2090.8.20")  The loop above will print out the string inside print() over and over again until the 20th of August, 2090.   * A list comprehension is an expression that creates a list by iterating over another container. * A **basic** list comprehension:   [i\*2 for i in [1, 5, 10]]  Output: [2, 10, 20]   * List comprehension with **if** condition:   [i\*2 for i in [1, -2, 10] if i>0]  Output: [2, 20]   * List comprehension with an **if** **and** **else** condition:   [i\*2 if i>0 else 0 for i in [1, -2, 10]]  Output: [2, 0, 20]   * Functions can have more than one **parameter**:   def volume(a, b, c):  return a \* b \* c   * Functions can have **default** parameters (e.g. coefficient):   def converter(feet, coefficient = 3.2808):  meters = feet / coefficient  return meters  print(converter(10))  Output: 3.0480370641306997  Arguments can be passed as **non-keyword** (positional) arguments (e.g. a) or **keyword** arguments (e.g. b=2 and c=10):  def volume(a, b, c):  return a \* b \* c  print(volume(1, b=2, c=10))   * An **\*args** parameter allows the  function to be called with an arbitrary number of non-keyword arguments:   def find\_max(\*args):  return max(args)  print(find\_max(3, 99, 1001, 2, 8))  Output: 1001   * An **\*\*kwargs** parameter allows the function to be called with an arbitrary number of keyword arguments:   def find\_winner(\*\*kwargs):  return max(kwargs, key = kwargs.get)  print(find\_winner(Andy = 17, Marry = 19, Sim = 45, Kae = 34))  Output: Sim   * can **read** an existing file with Python:   with open("file.txt") as file:  content = file.read()   * can **create** a new file with Python and **write** some text on it:   with open("file.txt", "w") as file:  content = file.write("Sample text")   * can **append** text to an existing file without overwriting it:   with open("file.txt", "a") as file:  content = file.write("More sample text")   * Can do both **append and read** a file with:   with open("file.txt", "a+") as file:  content = file.write("Even more sample text")  file.seek(0)  content = file.read()   * **Builtin objects** are all objects that are written inside the Python interpreter in C language. * **Builtin modules** contain builtins objects. * Some builtin objects are not immediately available in the global namespace. They are parts of a builtin module. To use those objects the module needs to be **imported** first. E.g.: * import time   time.sleep(5)   * **A list of all builtin modules** can be printed out with: * import sys   sys.builtin\_module\_names   * **Standard libraries** is a jargon that includes both builtin modules written in C and also modules written in Python. * **Standard libraries** written in Python reside in the Python installation directory as *.py* files. You can find their directory path with sys.prefix. * **Packages** are a collection of *.py* modules. * **Third-party libraries** are packages or modules written by third-party persons (not the Python core development team). * Third-party libraries can be **installed** from the terminal/command line:   Windows:  pip install pandas  Mac and Linux:  pip3 install pandas  ***Tcs ion***   |  |  | | --- | --- | | ***Topic:*** | ***1.Ace corporate interview 2.Learn corporate etiquette***  ***3.write effective emails*** |   ***Image of session***        **Report**  **Ace corporate interviews**   * What id an interview? * Interview process * Preparation for an effective interview include:  1. A good assessment of yourself 2. Researching the organization 3. Updating your resume 4. Understanding the venue details  * Do’s before the interview  1. Dress appropriately as per the corparate setting 2. Take care of pesonal grooming and cleanliness 3. Reach 10-15 minutes early  * Don’ts before an interview  1. Don’t stay up late at night 2. Don’t feel nervous 3. Don’t forget to be courteous to everybody  * Mens interview attire * Women interview attire * During the interview Do’s  1. Ask for clarification if you don’t understand question 2. Be brief and concise in your response  * During the interview Don’ts  1. Don’t take a seat until you are offered one 2. don’t slouch and fidget  * Tell something about yourself * Why shoud we hire you? * What are your strenght? * What are your weaknesses? * What are your achievements? * What is your career objective?   **Learn corporate etiquette**     * Basic rules-courtesies-Business cards * Basic rules-courtesies-space * Basic rules-courtesies-In meeting * Basic rules-courtesies-Language * Basic rules-Personal details * Basic rules-Personal hygiene * Basic rules-At the cafeteria * Basic rules to be followed  1. Interruptig a person 2. Dressing in business  * Cubicle etiquete * Internet etiquete * Meeting etiquete * Courtesies at the door and elevator   **Write effective emails**   * Describe the structure of a email * Develop an effective subject line and text * Utilize a few opening and closing phrases * State the do’s and dont’s of email writing * Draft an email using the pointers taught in the session * Do’s of email etiquette   1.Use strong subject line  2.Keep your email short  3.Type the correct email id  4.Reply within a reasonable time   * Don’ts of email etiquette  1. don’t use all upper case or all lower case 2. don’t use on word responses 3. don’t call as soon as you send the message | | | |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |