|  |
| --- |
| **INSY336 Data Handling and Coding for Analytics** |

1. **Build the program to ask the age of users and print out the message as follows:**

|  |
| --- |
| Output |
| How old are you? 23  I’m 23 years old |

|  |
| --- |
| **Code** |
| **age= input('How old are you?')**  **print("I'm", age, "years old")** |

1. **Generate program to take two strings from users and print out following messages.**

* The combined string by adding a comma(,) between strings.
* Find the length of the combined string and print the result.
* Check whether the combined string contains ‘a’ and print the result.

|  |
| --- |
| Output |
| Input first string: apple  Input second string: banana  Combined string: apple, banana  Length :12  Check\_a:True |

|  |
| --- |
| **Code** |
| **a=input('Input first string:')**  **b=input('Input second string:')**  **c=a+','+b**  **print('\nCombined String:', c, '\nlength:',len(c),'\nCheck a:', 'a' in c)** |

1. **Generate program to take two floating numbers from users and conduct arithmetic operation (+, \*, -, /). Print the results as follows**

|  |
| --- |
| Output |
| input the number: 3.0  input the number: 0.5  num1+num2: 3.5  num1-num2: 2.5  num1\*num2: 1.5  num1/num2: 6.0 |

|  |
| --- |
| **Code** |
| **a=float(input('Input the number:'))**  **b=float(input('Input the number:'))**  **print('\nnum1+num2:', a+b, '\nnum1-num2:',a-b,'\nnum1\*num2:', a\*b,'\nnum1/num2:', a/b)** |

1. **Generate a program to check if a number is divisible by 6.**

* Get an integer input from the user (say n)
* Check if n is divisible by 6.

(For a number to be divisible by 6, the n%2 and n%3 should be 0.)

|  |
| --- |
| **Code** |
| **a=int(input('Input the number:'))**  **if a%2==0 and a%3==0:**  **print('It is divisible by 6')**  **else:**  **print('It is not divisible by 6')** |

**5. What will be the output of the following code? Read the description and fill in the blank. (If you are not sure, try the code in the console)**

**String Method**

|  |  |
| --- | --- |
| Method | Description |
| **startwith()** | The **startswith()** returns a boolean. It returns True if the string **starts with** the specified prefix. It returns False if the string doesn't start with the specified prefix.    >>> a = 'Hello'  >>> a.startswith('He')   |  | | --- | | True |   >>> a.startswith('lo')   |  | | --- | | False | |
| **endswith()** | The **endswith**() returns True if a string **ends with** the specified suffix.  str.endswith(suffix)  >>> a = 'Hello'  >>> a.endswith('He')   |  | | --- | | False | |
| **find()** | The **find()** returns the lowest index of the substring if it is **found** in a given string. If its is not **found** then it returns to -1.  Parameters: It's the substring, which needs to be searched in the given string.  >>> a = 'Hello Hello'  >>> a.find('ll')   |  | | --- | | 2 |   >>> a.find('H')   |  | | --- | | 0 |   >>> a.find('K')   |  | | --- | | -1 | |
| **rfind()** | The **rfind**() method returns the last index to where the substring str is found, or -1 if no such index exists, optionally restricting the search to string[beg:end].  >>> a = 'Hello Hello'  >>> a.rfind('H')   |  | | --- | | 6 |   >>> a.rfind('lo')   |  | | --- | | 9 |   >>> a.rfind('M')   |  | | --- | | -1 | |
| **count()** | The **count()** method is an inbuilt function in Python that returns the count of how many times a given object occurs in a string.  >>> a = 'Hello'  >>> a.count('l')   |  | | --- | | 2 | |
| **lstrip()** | The **lstrip**() removes characters from the left based on the argument (a string specifying the set of characters to be removed).  >>> ' Left Strip'.lstrip()   |  | | --- | | Left Strip | |
| **rstrip()** | The **rstrip**() method returns a copy of the string with the trailing characters removed (based on the string argument passed). The **rstrip**() removes characters from the right based on the argument (a string specifying the set of characters to be removed)  >>> 'Right Strip '.rstrip()   |  | | --- | | 'Right Strip | |
| **strip()** | The **strip()** method returns a copy of the string with both leading and trailing characters removed (based on the string argument passed). The **strip()** removes characters from both left and right based on the argument (a string specifying the set of characters to be removed).  >>> ' Strip '.strip()   |  | | --- | | Strip | |
| **replace()** | The **replace()** is an inbuilt function in Python programming language that returns a copy of the string where all occurrences of a substring is replaced with another substring.  >>> a = 'Hello, World'  >>> b = a.replace('World', 'Korea')  >>> a   |  | | --- | | Hello, World |   >>> b   |  | | --- | | Hello, Korea | |
| **upper()** | The **upper()** method returns the uppercased string from the given string. It converts all lowercase characters to uppercase. If no lowercase characters exist, it returns the original string.  >>> a = 'lower case'  >>> b = a.upper()  >>> a   |  | | --- | | lower case | |
| **lower()** | The **lower()** method returns the lowercased string from the given string. It converts all uppercase characters to **lowercase**. If no uppercase characters exist, it returns the original string.  >>> a = 'UPPER CASE'  >>> b = a.lower()  >>> a   |  | | --- | | UPPER CASE |   >>> b   |  | | --- | | lower case | |
| **format** | The string formatting methods*allow multiple substitutions and value formatting. This method lets us concatenate on elements within a string through positional formatting.*  Formatters work by putting in one or more replacement fields and placeholders defined by a pair of curly brackets **{ }**into a string and calling the str.format(). The value we wish to put into the placeholders and concatenate with the string passed as parameters into the format function.  >>> a = 'My name is {0}. I am {1} years old.'.format('Mario', 40)  >>> a   |  | | --- | | My name is Mario. I am 40 years old. |   >>> b = 'My name is {name}. I am {age} years old.'.format(name='Luigi', age=35)  >>> b   |  | | --- | | My name is Luigi. I am 35 years old. | |
| **isalpha()** | The **isalpha**() method returns True if all characters in the string are alphabets. If not, it returns False.  >>> 'ABCDefgh'.isalpha()   |  | | --- | | True |   >>> '123ABC'.isalpha()   |  | | --- | | False | |
| **isalnum()** | The **isalnum()** method returns True if all characters in the string are alphanumeric(either alphabet or numeric) If not, it returns False.  >>> '1234ABC'.isalnum()   |  | | --- | | True |   >>> '1234'.isalnum()   |  | | --- | | True | |
| **split()** | **The split()** method returns a list of strings after breaking the given string by the specified separator.  *str.split(separator)*  ***separator :****This is a delimiter. The string splits at this specified separator. If is not provided then any white space is a separator.*  >>> ﻿sentence='welcome to INSY336'  >>> ﻿sentence.split(' ')   |  | | --- | | ['welcome’, ‘to', 'INSY336'] |   ﻿>>> sentence='welcome, to, INSY336'  >>> sentence.split(',')     |  | | --- | | ['welcome’, ‘to', 'INSY336'] | |