1. Activity 1: Errors

Incorporate the following code into a Python program to handle exceptions.

```
try:
  # do something
  pass

except ValueError:
  # handle ValueError exception
  pass

except (TypeError, ZeroDivisionError):
  # handle multiple exceptions
  # TypeError and ZeroDivisionError
  pass

except:
  # handle all other exceptions
  Pass
```

Source: Programiz (n.d.) **Python Exception Handling Using try, except and finally statement**.

Python can encounter exceptions during program execution, such as SyntaxError, TypeError, NameError, IndexError, KeyError, ValueError, AttributeError, IOError, ZeroDivisionError, and ImportError. These exceptions have specific meanings, such as syntax errors, type errors, variable or function name not found, index out of range, key not found in a dictionary, invalid arguments, missing attributes/methods, input/output errors, and module import failures (GeeksforGeeks, 2016).

I have included the entire Python program code for managing exceptions, using GeeksforGeeks' (2021) examples as a reference:

```
def factorial(n):
    if n < 0:
        raise ValueError("n needs to be positive.")
    if n == 0:
        return 1
```

```
    return n * factorial(n - 1)

def main():
    try:
        # do something that may raise an exception
        print(factorial(-1))

    except ValueError:
        # handle ValueError exception
        print("ValueError: n must be non-negative")

    except (TypeError, ZeroDivisionError):
        # handle multiple exceptions
        # TypeError and ZeroDivisionError
        print("TypeError or ZeroDivisionError")

    except:
        # handle all other exceptions
        print("Unknown exception")

if __name__ == "__main__":
    main()
```

In this program, the first function defined is factorial(). It takes an integer as input and returns its factorial. If the input value is less than 0, a ValueError exception is raised. If the input is 0, the function returns 1. Otherwise, the function recursively calculates the factorial of the input.

In the program's main function, the factorial() function is called with negative input, which triggers the ValueError exception. The except clause for ValueError will then print the relevant error message.

If the input provided to the factorial() function is either not an integer or is 0; it will lead to a TypeError or ZeroDivisionError exception. Moreover, if the second argument passed to the factorial() function is a negative number; it may cause issues. In such cases, the unless clause will print an error message appropriate for the specific error, i.e., TypeError or ZeroDivisionError.

If exceptions occur, the except clause using the * wildcard will handle them and display the relevant error message.

2. Activity 2: Data Structures

2.1 Set operations include Union, Intersection, Difference, and Symmetric difference. Explain a use for each of these set operations within the context of your summative assessment.

Within the context of the above code, here are some examples of how to employ set operations such as union, intersection, difference, and symmetric difference:

Union: When two sets are merged, their union contains all elements from one or both groups. To illustrate, when the sets 1, 2, 3 and 2, 4, 5 are combined, their union is 1, 2, 3, 4, 5 (Vinci, 2019).

Similarly, the union of the sets of all positive and even integers equal the set of all integers. This is because every positive integer is either even or odd, and every actual integer is likewise positive.

Intersection: When two sets have common elements, their intersection is the collection of these elements. To illustrate, the intersection of {1, 2, 3} and {2, 4, 5} is {2} (Vinci, 2019).

In the previous code, when the set of all positive integers intersects with all even integers, the resulting set is the set of all even integers. This is because only even integers can be both positive and.

Difference: The collection of components in the first set that is not in the second set distinguishes the two groups. For example, sets 1 and 3 differ between sets 1, 2, 3 and 2, 4, 5 (Vinci, 2019).

In the context of the preceding code, the set of all odd integers is the difference between the set of all positive numbers and all even integers. This is because the only positive integers that are not even integers are the odd integers.

Symmetric difference: The symmetric difference between two sets is the combined group of elements that exist in only one of the sets, not both. For example, if we consider the sets 1, 2, 3 and 2, 4, 5, their symmetric difference is 1, 3, 4, 5 (Vinci, 2019).

In the previous code, the symmetric difference between the sets containing all positive numbers and all even integers includes all odd integers and even integers that are not 2. This is because the only odd integers are non-even positive integers, and the superior even numbers not two are those greater than 2.

2.2 Write a Python program to conduct a linear search on a list data structure.

Linear search is a sequential searching strategy in which we start at one end of the list and verify each element until the desired element is located. It is the most basic search algorithm (Programiz, N.D.).

For instance, here is my code:

```
def linear_search(list, item):
    for i in range(len(list)):
        if list[i] == item:
            return i
    return -1

def main():
    list = [1, 2, 3, 4, 5]
    item = 3
    index = linear_search(list, item)
    if index == -1:
        print("Item not found")
    else:
        print("Item found at index", index)

if __name__ == "__main__":
    main()
```

This programme uses the linear_search() method to find a specific item in a list. The function takes two inputs, a list and an item, and outputs the index of the discovered item. If the item cannot be located, the function returns -1. The process loops through

the list while comparing each element and the object. The method returns the element's index if the element and the item match.

The program's main function uses a list and an item as inputs for the linear_search() method. The method returns the item's index if it has been found. If the item cannot be located, the function returns -1. The main process generates a message indicating whether or not the object was located.

References:

GeeksforGeeks. (2016). *Python Exception Handling*. [online] Available at: https://www.geeksforgeeks.org/python-exception-handling/.

GeeksforGeeks. (2021). *Python - Catch All Exceptions*. [online] Available at: https://www.geeksforgeeks.org/python-catch-all-exceptions/ [Accessed 2 Jul. 2023].

Vinci, S. G. (2019). *Intersection, union and difference of Sets in Python.* [online] Available at: https://dev.to/svinci/intersection-union-and-difference-of-sets-in-python-4gkn#:~:text=Union%3A%20All%20the%20elements%20from [Accessed 2 Jul. 2023].

www.programiz.com. (n.d.). *Linear Search*. [online] Available at: https://www.programiz.com/dsa/linear-search.