

Segales Ace Niño B.

BSCPE 2A

Laboratory Exercise No. 2 CH2

Title: Understanding Programming Constructs

Brief Introduction

Programming constructs are the building blocks of any software application. These include variables, loops, conditionals, functions, data structures, and recursion. This lab focuses on these constructs and their implementation in Python.

Objectives

- Learn and implement basic programming constructs.
- Explore recursion with practical examples.

Detailed Discussion

Variables: Variables are containers for storing data values. In Python, variables do not need explicit declaration and can hold data of any type.

Loops: Loops are used to execute a block of code repeatedly. Python supports for and while loops.

Conditional Statements: These allow branching based on conditions. Python uses if, elif, and else for decision-making.

Functions: Functions are reusable blocks of code that perform specific tasks. They can accept inputs (parameters) and return outputs.

Data Structures: Python supports several built-in data structures like lists, dictionaries, sets, and tuples to organize and manage data efficiently.

Recursion: Recursion is a technique where a function calls itself to solve a problem. It is useful for tasks like calculating factorials, generating Fibonacci sequences, and traversing data structures.

Detailed Discussion

Programming Paradigm	Description	Example Applications
Imperative	Focuses on step-by-step instructions.	Low-level programming tasks
Object-Oriented	Organizes code using objects and classes.	GUI applications, simulations
Functional	Emphasizes mathematical functions and immutability.	Data analysis, AI
Declarative	Specifies what to do without describing how to do it.	SQL, configuration files
Event-Driven	Responds to events like clicks, signals, or messages.	GUIs, games
Concurrent	Manages multiple computations at the same time.	Web servers, parallel processing

Materials

- Computer system with Python.
- VS Code IDE.

Time Frame

1.5 hours

Procedure

1. Create a new Python file in VS Code.
2. Implement examples of variables, loops, conditionals, and recursion.

Example Code:

```
# Variables and loops
nums = [1, 2, 3, 4, 5]
for num in nums:
    print(num)
```

```
# Conditional Statements
```

```
if len(nums) > 3:
```

```
    print("List is large")
```

```
else:
```

```
    print("List is small")
```

```
# Recursion
```

```
def factorial(n):
```

```
    if n == 0:
```

```
        return 1
```

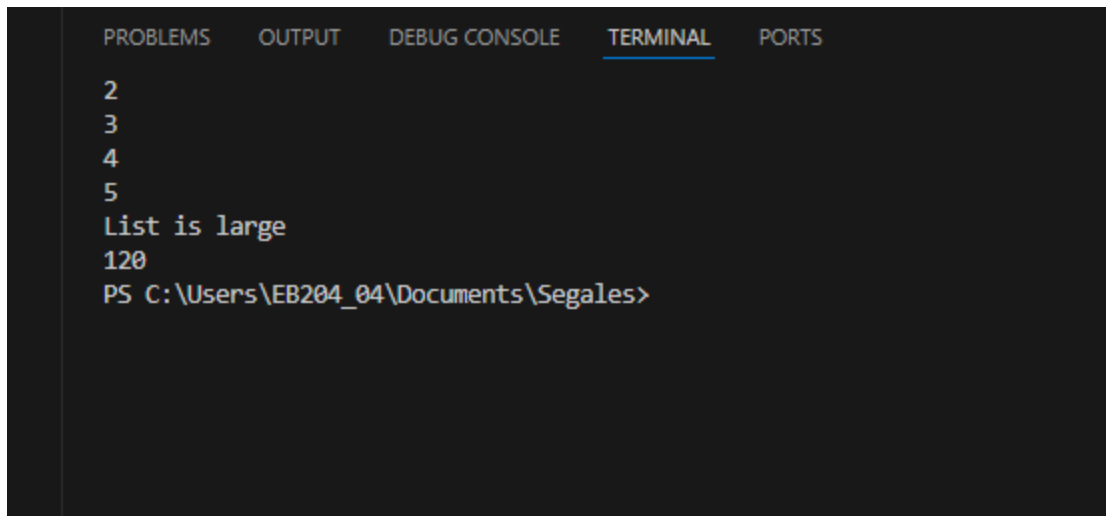
```
    return n * factorial(n-1)
```

```
print(factorial(5))
```

Run the program and observe the results.

lab exercise 2 > ...

```
1  # Variables and loops
2
3  nums = [1, 2, 3, 4, 5]
4
5  for num in nums:
6      |
7      print(num)
8
9
10
11 # Conditional Statements
12
13 if len(nums) > 3:
14     |
15     print("List is large")
16
17 else:
18     |
19     print("List is small")
20
21
22
23 # Recursion
24
25 def factorial(n):
26     |
27     if n == 0:
28         |
29         return 1
30
31     return n * factorial(n-1)
32
33
34
35 print(factorial(5))
```

A screenshot of a terminal window with a dark background. At the top, there are five tabs: 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL' (which is selected and underlined), and 'PORTS'. The terminal displays the following text: '2', '3', '4', '5', 'List is large', '120', and a command prompt 'PS C:\Users\EB204_04\Documents\Segales>'.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
2
3
4
5
List is large
120
PS C:\Users\EB204_04\Documents\Segales>
```

Results

Record outputs for different test cases.

Follow-Up Questions

1. What is the purpose of using recursion?

Recursion is used for code simplification, efficient issue solving, work division into manageable chunks, state management, and the provision of elegant solutions to recursive problems. Recursion should be used sparingly, though, as improper implementation with base cases and limitations might result in higher memory utilization and possible stack overflow issues.

2. How do loops differ from recursion?

The methods of repetition, state management, termination criteria, memory utilization, performance, and common use cases are different between loops and recursion. The decision between the two constructions frequently depends on the particular problem being handled as well as the required level of clarity and efficiency of the solution, even if both can produce results that are comparable.

3. Explain a scenario where conditionals are essential.

When decisions must be made based on certain standards or conditions, conditionals are crucial. User authentication and access control in online applications are two typical contexts where conditionals are essential.

Consider an online program that contains capabilities like reading personal information, purchasing, or accessing premium content that require users to log in.

The application compares the user's credentials (password and username) with the stored information when the user tries to log in.

Based on the user's role (e.g., admin, regular user, guest), the program must decide which features the user can access after logging in.

Findings

Demonstrate understanding of the constructs through written observations.

Summary

Programming constructs provide the foundation for problem-solving. Students implemented basic constructs effectively.

Conclusion

By understanding these constructs, students are equipped to tackle programming challenges and design efficient algorithms.