



## IT Midterm Study Guide - 10th grade

This guide covers the key concepts you need to know for your exam, from Python basics to control flow and user input.

### Part 1: Python Fundamentals

#### 1. Variables and Data Types

- **Variables** are used to store data in memory. Using variables makes code simpler and reduces repetition.
- **Rules for Naming Variables:**
  - Names can only contain letters, numbers, and underscores.
  - They can start with a letter or an underscore, but not a number.
  - Spaces are not allowed, but you can use underscores to separate words.
  - Avoid using Python keywords (like `if`, `for`, etc.) and built-in function names (like `print`).
  - Names should be short but descriptive (e.g., `name` is better than `n`).
- **Data Types:** Python has several data types, including Numbers, Strings, and Lists. You can check a variable's type using the built-in `type()` function.

#### 2. Strings

- A string is a series of characters inside single or double quotes.
- **Common String Methods:**
  - `.upper()` and `.lower()`: Convert a string to all uppercase or all lowercase. These methods do not change the original string variable unless you reassign it.
  - `.title()`: Converts the first letter of each word to uppercase.
  - `.strip()`, `.lstrip()`, `.rstrip()`: Remove extra whitespace from both ends, the left end, or the right end of a string.

- **Formatting Strings:**
  - **f-strings** (available in Python 3.6+) let you embed variables directly into a string. You start the string with `f` and put variable names in curly braces `{}`.
  - **Whitespace:** You can add a tab with `\t` and a newline with `\n` to organize output.

### 3. Numbers

- **Types:** Python handles integers (like `1`, `2`) and floats (numbers with a decimal point, like `1.2`).
- **Operations:** You can perform addition (+), subtraction (-), multiplication (\*), and division (/). Use two asterisks (\*\*) for exponents.

## Part 2: Control Flow

### 1. Conditional Logic (If Statements)

- **Conditional Tests:** A conditional test is an expression that evaluates to `True` or `False`. These are also known as Boolean expressions.
- **Comparison Operators:**
  - `==` (equals) vs. `=` (assignment).
  - `!=` (not equal).
  - `>`, `>=`, `<`, `<=` (numerical comparisons).
  - Use the `.lower()` method on strings to ignore case when checking for equality.
- **Multiple Conditions:**
  - `and`: Returns `True` only if all conditions are `True`.
  - `or`: Returns `True` if at least one condition is `True`.
- **Checking Membership:** Use `in` and `not in` to check if a value is present in a list.
- **If Structures:**
  - `if`: Executes a block of code if the test is `True`.

- **if-else:** Executes one block of code if the test is **True** and a different block if it is **False**.
- **if-elif-else:** A chain used to test multiple conditions in order. Python runs each test until one passes, executes that block, and skips the rest.
- **Using if with Lists:** You can check if a list is empty. An **if** statement on a list will evaluate to **True** if the list contains items and **False** if it is empty.

### Part 3: User Interaction

- **Getting Input:** The **input()** function is used to get input from a user. It displays a message (prompt) and waits for the user to type something and press Enter.
- **Handling Input:**
  - The **input()** function always returns a string.
  - To treat the input as a number, you must convert it using **int()** (for an integer) or **float()** (for a decimal).