C, C++ & Python Comparative Basic Tutorial

A tutorial for getting started with C, C++, and Python, focusing on installation and comparative examples.

Task 1: Hello, World! (Basic I/O)

This is the classic first program. It just prints a line of text.

C

C uses the stdio.h (standard input/output) library and its printf function.

C

```c
#include <stdio.h>

int main() {
    printf("Hello, World!\n");   // "\n" indicates new line
    return 0;
}
```

- **To Compile/Run:**

1. gcc hello.c -o hello
2. ./hello (or hello.exe on Windows)

C++

C++ uses the iostream library and the cout (character output) stream.

C++

```cpp
#include <iostream>

using namespace std;

int main() {
    cout << "Hello, World!" << endl;  // “endl” indicates new line
    return 0;
}
```

- **To Compile/Run:**

1. g++ hello.cpp -o hello
2. ./hello (or hello.exe on Windows)

Python

Python uses the built-in print() function. No compiler is needed.

Python

```python
print("Hello, World!")
```

- **To Run:**

1. python3 hello.py

Task 2: Variables & Arithmetic (Integers & Floats)

Here we declare numbers, perform a calculation, and print the result.

C

C is **statically typed**, so you must declare the type (int, float) of each variable before using it.

C

```c
#include <stdio.h>

int main() {
   int a = 10;
   float b = 4.5;
   float sum = a + b;

   // "%d" is a format specifier for integer numbers
   // "%f" is a format specifier for float/decimal numbers
   printf("The sum of %d and %f is %f\n", a, b, sum);

   return 0;
}
```

C++

C++ is also **statically typed**, but its iostream library automatically handles formatting.

C++

```cpp
#include <iostream>

using namespace std;

int main() {
   int a = 10;
   float b = 4.5;
   float sum = a + b;

   cout << "The sum of " << a << " and " << b << " is " << sum << endl;

   return 0;
}
```

Python

Python is **dynamically typed**. You don't declare types; the variable's type is determined at runtime.

Python

```python
a = 10
b = 4.5
sum = a + b

# You can print variables directly or use an f-string
```

```
print(f"The sum of {a} and {b} is {sum}")
```

Task 3: User Input (Strings & Type Conversion)

Ask the user for their name and age and print a message.

C

Handling strings in C is complex, requiring a character array. scanf is used for input, which can be tricky.

C

```c
#include <stdio.h>

int main() {
    char name[50]; // Allocate 50 bytes for a string, each byte can contain a single character
    int age;

    printf("Enter your name: ");
    // Read a line of text, careful with buffer
    fgets(name, 50, stdin);

    printf("Enter your age: ");
    scanf("%d", &age); // Read an integer

    // Note: fgets includes the newline, we'd need to remove it
    printf("Hello, %s. You are %d years old.\n", name, age);

    return 0;
}
```

C++

C++ simplifies this immensely with the string type and the cin stream.

C++

```cpp
#include <iostream>
#include <string> // Need this for the string type

using namespace std;

int main() {
    string name;
    int age;

    cout << "Enter your name: ";
    getline(cin, name); // Reads a full line of text

    cout << "Enter your age: ";
    cin >> age; // Reads an integer

    cout << "Hello, " << name << ". You are " << age << " years old." << endl;
```

```
    return 0;
}
```

Python

Python's input() function makes this trivial. All input is received as a string, so you must **cast** the age to an int.

Python

```
name = input("Enter your name: ")
age_str = input("Enter your age: ")

# Convert the string "age_str" to an integer
age = int(age_str)

print(f"Hello, {name}. You are {age} years old.")
```

Task 4: Arrays & Lists (Collections)

Store a list of numbers and loop through them.

C

C uses **fixed-size arrays**. You must know the size when you declare it.

C

```
#include <stdio.h>

int main() {
    // Array must have a fixed size
    int numbers[4] = {10, 20, 30, 40};
    int i;

    for (i = 0; i < 4; i++) {
        printf("Element %d: %d\n", i, numbers[i]);
    }

    return 0;
}
```

C++

C++ has fixed-size arrays like C, but its standard library provides a more flexible vector (a dynamic array).

C++

```
#include <iostream>
#include <vector> // Need this for vectors

using namespace std;

int main() {
```

```
    // A vector can grow or shrink
    vector<int> numbers = {10, 20, 30, 40};

    // Add a new element
    numbers.push_back(50);

    int i;

    for (i = 0; i < numbers.size(); i++) {
        cout << "Element " << i << ": " << numbers[i] << endl;
    }

    return 0;
}
```

Python

Python's list is a powerful, dynamic collection that can hold items of any type.

Python

```
# A list is dynamic and flexible
numbers = [10, 20, 30, 40]

# Add a new element
numbers.append(50)

# You can even mix types
numbers.append("sixty")

for num in numbers:
    print(f"Element: {num}")
```

Task 5: Functions

Define a simple function that takes two numbers and returns their sum.

C

Functions must be declared with explicit return and argument types.

C

```
#include <stdio.h>

// Function definition
int add(int x, int y) {
    return x + y;
}

int main() {
    int result = add(5, 3);
    printf("The result is %d\n", result);
```

```
    return 0;
}
```

C++

Syntax is identical to C for this simple case.

C++

```
#include <iostream>

using namespace std;

// Function definition
int add(int x, int y) {
    return x + y;
}

int main() {
    int result = add(5, 3);
    cout << "The result is " << result << endl;
    return 0;
}
```

Python

Use the def keyword to define a function. No types are declared.

Python

```
# Function definition
def add(x, y):
    return x + y

result = add(5, 3)
print(f"The result is {result}")

# Python's dynamic typing lets it work with floats too
result_float = add(5.5, 2.1)
print(f"The float result is {result_float}")
```

These are the basics that will help you to start coding for this course in any of the 3 languages. However, there are many other advanced things to learn for various other use cases. You can learn further from these resources:
C: https://www.w3schools.com/c/
C++: https://www.w3schools.com/cpp/
Python: https://www.w3schools.com/python/