Tutorial 17 - Inline Functions, Default Arguments, and Constant Arguments in C++

Inline Functions in C++

- **Definition**: Reduces function call overhead by expanding the function code at the call point.
- **Use Case**: Suitable for small functions called frequently.
- **Example**:

```
inline int product(int a, int b) {
    return a * b;
}

```

- **Characteristics**:
  - Avoid using `inline` with static variables.
  - Helps improve execution time for small, frequently called functions.

Program:

```
#include<iostream>
using namespace std;
inline int product(int a, int b) {
    return a * b;
}
int main() {
    int a, b;
    cout << "Enter the value of a and b: ";
    cin >> a >> b;
    cout << "The product of a and b is " << product(a, b) << endl;
    cout << "The product of a and b is " << product(a, b) << endl;
    return 0;
}

```

Default Arguments in C++

- **Definition**: Predefined values for function parameters if arguments are not provided during the call.
- **Rules**:
  - Default arguments must be placed after non-default ones.
- **Example**:

```
float moneyReceived(int currentMoney, float factor = 1.04) {
    return currentMoney * factor;
}

```

Program:

```
#include<iostream>
using namespace std;
float moneyReceived(int currentMoney, float factor = 1.04) {
    return currentMoney * factor;
```

```
}
int main() {
    int money = 100000;
    cout << "If you have " << money << " Rs in your bank account, you will receive "
         << moneyReceived(money) << " Rs after 1 year" << endl;
    cout << "For VIP: If you have " << money << " Rs in your bank account, you will receive "
         << moneyReceived(money, 1.1) << " Rs after 1 year" << endl;
    return 0;
}

```

Constant Arguments in C++

- **Definition**: Prevents modification of argument values within a function.
- **Usage**: Use `const` keyword before the parameter type.
- **Example**:

```
int strlen(const char *p) {
    // Argument cannot be modified
    return 0;  // Placeholder
}

```

Program:

```
#include<iostream>
using namespace std;
int strlen(const char *p) {
    // Logic to calculate string length without modifying `p`
    return 0;  // Placeholder
}
int main() {
    const char *str = "Hello";
    cout << "String length is: " << strlen(str) << endl;
    return 0;
}

```

Key Points for Notebook

1. **Inline Functions**:
   - Use `inline` to reduce overhead of function calls.
   - Suitable for small functions.
   - Example: `inline int product(int a, int b) { return a * b; }`
2. **Default Arguments**:
   - Provides default parameter values.
   - Defaults are used if arguments are not passed.
   - Example: `float moneyReceived(int a, float b = 1.04);`
3. **Constant Arguments**:
   - Prevents parameter modification.
   - Use `const` keyword before parameter type.
   - Example: `int strlen(const char *p);`