Validating User Input

Ensuring Input Integrity

Basic Input Validation in Go

Use `fmt.Scan` and error handling to validate input in Go.

```
var age int

fmt.Print("Enter your age: ")
_, err := fmt.Scan(&age)

if err != nil || age < 0 {
    fmt.Println("Invalid input. Please enter a valid age.")
}
```

Basic Input Validation in Python

Python's `try-except` for input validation.

```python
try:
    age = int(input("Enter your age: "))
except ValueError:
    print("Invalid input. Please enter a valid age.")
```

Basic Input Validation in C++

C++ input validation using `cin` and conditional checks.

```cpp
#include <iostream>
using namespace std;

int main() {
    int age;

    cout << "Enter your age: ";
    if (!(cin >> age) || age < 0) {
        cout << "Invalid input. Please enter a valid age." << endl;
    }
    return 0;
}
```

Loop-Based Input Validation in Go

Loop until valid input is provided in Go.

```go
var age int

for {
    fmt.Print("Enter your age: ")
    _, err := fmt.Scan(&age)

    if err == nil && age >= 0 {
        break
    }
    fmt.Println("Invalid input. Please enter a valid age.")
}
```

Loop-Based Input Validation in Python

Python's loop-based input validation.

```
while True:
    try:
        age = int(input("Enter your age: "))
        if age >= 0:
            break
    except ValueError:
        pass
    print("Invalid input. Please enter a valid age.")
```

Loop-Based Input Validation in C++

C++ loop-based input validation with `cin`.

```cpp
#include <iostream>
using namespace std;

int main() {
    int age;

    while (true) {
        cout << "Enter your age: ";
        if (cin >> age && age >= 0) {
            break;
        }
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
        cout << "Invalid input. Please enter a valid age." << endl;
    }

    return 0;
}
```

Summary

- Input validation ensures data integrity.
- GoLang, Python, and C++ provide methods for validation.
- Loop-based validation for repeated attempts.