Simplified Test Case Language (STCL)

Version 3.0

Test Framework Team

April 30, 2025

Contents

1 Introduction

1.1 What is STCL?

Simplified Test Case Language (STCL) is a domain-specific language designed for generating test cases for Competitive Programming and Data Structure/Algorithm problems.

1.2 Key Features

- Intuitive English-like syntax
- Smart default values
- Rich set of data structure generators
- Built-in constraint validation
- Competition-ready templates

2 Core Concepts

2.1 Basic Syntax

```
# Variable declaration template
var <name>: <type>(<parameters>);
```

2.2 Primitive Types

Type	Example	Description
int	var n: int(1..100);	Integer with optional range
float	var temp: float(0.0..1.0, 2);	Float with precision
char	var c: char(upper+number);	Character from set
bool	var flag: bool(0.7);	70% true probability

3 Collections

3.1 Arrays

```
var arr1: [int](10);                # Random integers
var arr2: [int](10, sorted=asc);    # Sorted ascending
var arr3: [int](10, unique);        # All unique values
```

3.2 Strings

```
var s1: string(10);                 # Random characters
var s2: string(7, palindrome);      # Palindromic string
var s3: string("a*b+c?d", regex);   # Regex-compliant
```

4 Graphs and Trees

4.1 Graph Types

```
var g1: graph(undirected, nodes=10, edges=15);
var g2: graph(dag, nodes=8, edges=12);
```

4.2 Tree Structures

```
var t1: tree(bst, nodes=10, range=1..100);
var t2: tree(max_heap, nodes=15);
```

5 Specialized Generators

5.1 Permutations

```
var p1: permutation(5);              # Random shuffle
var p2: permutation(8, derangement);# No fixed points
```

5.2 Matrix Generators

```
var m1: matrix[int](3x3);            # Random matrix
var m2: matrix[int](5x5, spiral);    # Spiral pattern
```

6 Problem Templates

6.1 Binary Search

```
template binary_search {
    n: int(1e5),
    arr: [int](n, sorted=asc, unique),
    target: arr[random(0..n-1)]
}
```

6.2 Dynamic Programming

```
template knapsack {
    capacity: int(1e3),
    weights: [int](20, 1..100),
    values: [int](20, 1..100)
}
```

7 Error Handling

7.1 Common Errors

Error	Resolution
Invalid range	Check min/max values
Constraint conflict	Verify parameter compatibility
Undefined variable	Ensure variable declaration order

8 Advanced Features

8.1 Batch Testing

```
var tests: batch(100, {
    n: int(1..1e5),
    arr: [int](n, sorted=asc)
});
```

8.2 Visual Debugging

```
debug maze; # Generates ASCII + SVG visualization
```

A Quick Reference

A.1 Cheatsheet

Construct	Example
Array	var arr: [int](10, sorted=asc);
Matrix	var mat: matrix[int](5x5, spiral);
Graph	var g: graph(dag, nodes=10, edges=15);
String	var s: string(10, palindrome);