Random Test Data Generation
Second Software Executable

Started 02/03/2026
Maksym Redchenko C00302240
SETU

1. Delivery Format

The software is delivered as a VS Code Extension packaged in the standard .vsix format — a self-contained, platform-independent installer that works on Windows, Linux, macOS, and WSL.

Packaged file: s5in-c-bva-test-generator-2.0.3.vsix

2. Installation Instructions

Step 1: Install Visual Studio Code

Download and install VS Code from https://code.visualstudio.com if not already installed.

Step 2: Install the Extension

Via VS Code UI:

1. Open VS Code
2. Open the Extensions sidebar (Ctrl+Shift+X)
3. Search for "S5in C BVA Test Generator"
4. Click Install on the extension by S5in
5. The extension activates immediately — no reload required

Step 3: Install Runtime Prerequisites

The extension itself requires no additional dependencies to install and activate. However, to build and execute the generated Google Test test cases, the following tools must be available on the host system:

Tool	Minimum Version	Purpose
C/C++ Compiler (g++ or clang++)	Any recent version	Compiles the generated test files
CMake	≥ 3.14	Builds the test project
Google Test (GTest)	Any recent version	Testing framework used by generated tests

Ubuntu / Debian / WSL:

```
    sudo apt update
    sudo apt install -y build-essential cmake libgtest-dev
    cd /usr/src/gtest && sudo cmake . && sudo make && sudo cp lib/*.a
/usr/lib/
```

macOS:

```
  brew install cmake googletest
```

Verify installation:

```
g++ --version     # Should display compiler version
cmake --version   # Should display version ≥ 3.14

ls /usr/lib/libgtest*.a 2>/dev/null || ls /usr/local/lib/libgtest*.a 2>/dev/null
                  # Should list libgtest.a and libgtest_main.a
```

3. How to Run the Software

- Open any .c source file in VS Code
- Place the cursor inside a C function
- Trigger test generation using one of the following methods:
    - Keyboard shortcut: Ctrl+Shift+T (Cmd+Shift+T on macOS)
    - Right-click context menu: Select “Generate Tests for This Function”
    - Command Palette: Ctrl+Shift+P → type “C Test Generator: Generate Tests for This Function”
- The extension opens the Expected Values webview panel with three tabs: Boundary Tests, Custom Tests, and Preview
- Fill in expected values for each boundary case, add custom tests if needed, and preview the generated code
- Click “Save & Build & Run” to compile and execute the tests — results appear in the VS Code terminal with colour-coded pass/fail output
- Build errors are surfaced in the VS Code Problems panel with clickable file and line information

4. What’s New in Version

Iteration 2 delivered five major goals beyond the Iteration 1 foundation:

Goal	Feature	Details
1	Pointer, Array & Struct support	int* (NULL + valid-pointer tests), int[] with size pairs, struct with zero-init + extreme-value tests
2	Smarter Boundary Values	float/double: ±FLT_MAX, FLT_EPSILON; char: '\0' and 'A'; size_t: 0, 1, 10, SIZE_MAX; EXPECT_FLOAT_EQ / EXPECT_DOUBLE_EQ assertions corrected

3	Test Density Configuration	New cTestGenerator.testDensity setting with minimal, standard (default), and exhaustive modes
4	UI Improvements	3-tab webview (Boundary Tests, Custom Tests, Preview); per-field struct inputs; Select/Deselect All; stats bar; syntax-highlighted preview
5	Better Error Reporting	Build errors in VS Code Problems panel (clickable, file + line); dedicated Output Channel logging; clearer error messages

5. Portability Statement

This software does not run “just on my laptop.” It is distributed as a standard .vsix installer that can be installed on any machine running VS Code. The following properties confirm its portability:

- No source code compilation required — the extension is pre-bundled (TypeScript compiled to JavaScript, WASM binaries included)
- No Node.js or npm required — all dependencies are packaged inside the .vsix
- No internet connection required — the extension works fully offline after installation
- Self-contained — the .vsix file contains everything needed to run the extension

The only external requirements (C++ compiler, CMake, Google Test) are standard development tools that are prerequisites of the problem domain — C test generation — not of the extension itself.

6. Reproducible Build

To rebuild the .vsix from source (for verification purposes):

```
git clone https://github.com/S5in/C_Random_Test_Data_Generation
  cd C_Random_Test_Data_Generation
  npm install
  npm install -g @vscode/vsce
  vsce package
```

This produces an identical .vsix package.

7. Iteration Versioning

Iteration	Version	File
Iteration 1	1.0.0	random-test-data-generation-1.0.0.vsix
Iteration 2	2.0.0	random-test-data-generation-2.0.3.vsix
Iteration 3	3.0.0	(future release)