README.md

Running the Bowling Score Calculator in VSCode

This guide will walk you through the steps to run the provided C++ Bowling Score Calculator code using Visual Studio Code (VSCode).

Prerequisites

1. **Visual Studio Code**: Make sure you have VSCode installed. You can download it from here.
2. **C++ Compiler**: Install a C++ compiler. For Windows, you can use MinGW or Microsoft Visual C++ Build Tools. For macOS, you can use Xcode Command Line Tools. For Linux, you can use `g++`.
3. **C++ Extension for VSCode**: Install the C++ extension by Microsoft for better C++ support in VSCode.

Steps to Run the Code

1. Open VSCode

Launch Visual Studio Code on your computer.

2. Install the C++ Extension

1. Go to the Extensions view by clicking the Extensions icon in the Activity Bar on the side of the window or press `Ctrl+Shift+X`.
2. Search for "C++" and install the extension named **C++** by Microsoft.

3. Create a New Project Folder

1. Create a new folder for your project. For example, name it `BowlingScoreCalculator`.
2. Open VSCode and go to `File` > `Open Folder...` and select the folder you just created.

4. Create and Save the C++ File

1. In VSCode, go to `File` > `New File` or press `Ctrl+N` to create a new file.
2. Copy and paste the provided C++ code into this new file.
3. Save the file with a `.cpp` extension, for example, `bowling_score_calculator.cpp`.

5. Create a Tasks File for Build Configuration

1. Go to the Explorer view by clicking the Explorer icon in the Activity Bar or press `Ctrl+Shift+E`.

2. Create a new folder in your project directory named `.vscode`.

3. Inside the `.vscode` folder, create a file named `tasks.json`.

4. Add the following content to `tasks.json`:

```
{
    "tasks": [
        {
            "type": "cppbuild",
            "label": "C/C++: g++.exe build active file",
            "command": "C:\\msys64\\mingw64\\bin\\g++.exe",
            "args": [
                "-fdiagnostics-color=always",
                "-g",
                "${file}",
                "-o",
                "${fileDirname}\\${fileBasenameNoExtension}.exe"
            ],
            "options": {
                "cwd": "${fileDirname}"
            },
            "problemMatcher": [
                "$gcc"
            ],
            "group": {
                "kind": "build",
                "isDefault": true
            },
            "detail": "Task generated by Debugger."
        }
    ],
    "version": "2.0.0"
}
```

5. Add the following content to `launch.json`:

```
{
    // Use IntelliSense to learn about possible attributes.
    // Hover to view descriptions of existing attributes.
    // For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387

    "version": "0.2.0",
    "configurations": [
        {
            "name": "C++ Launch",
            "type": "cppdbg",
            "request": "launch",
            "program": "${workspaceFolder}/bowling",
            "args": [],
            "stopAtEntry": false,
            "cwd": "${workspaceFolder}",
            "environment": [],
            "externalConsole": false,
            "MIMode": "gdb",
            "setupCommands": [
                {
                    "description": "Enable pretty-printing for gdb",
                    "text": "-enable-pretty-printing",
                    "ignoreFailures": true
                }
            ],
            "preLaunchTask": "build",
            "miDebuggerPath": "/usr/bin/gdb",
            "setupCommands": [
                {
                    "description": "Enable pretty-printing for gdb",
                    "text": "-enable-pretty-printing",
                    "ignoreFailures": true
                }
            ],
            "miDebuggerArgs": "",
            "logging": {
                "trace": true,
                "traceResponse": true,
                "engineLogging": true
            },
            "internalConsoleOptions": "openOnSessionStart"
        }
    ]
}
```

6. Build and Run the Code

1. Open the integrated terminal in VSCode by going to `Terminal` > `New Terminal` or pressing `Ctrl+`.

2. To build and run the code, press `Ctrl+Shift+B` or go to `Terminal` > `Run Build Task...` and select the `Build and Run` task.

    - This will compile the `bowling_score_calculator.cpp` file into an executable named `bowling_score_calculator`.
    - After successful compilation, run the executable by typing `./bowling_score_calculator` on **macOS/Linux** or `bowling_score_calculator.exe` on **Windows** in the terminal and press `Enter`.

7. Input Data and View Results

1. Follow the prompts in the terminal to enter the number of pins knocked down in each throw.

2. The program will display the results, including frame numbers, throws, and cumulative scores.

Troubleshooting

- `Compiler Not Found`: Ensure that the C++ compiler is installed and properly added to your system's PATH. You can test this by running `g++ --version` in your terminal.

- `Build Errors`: Check the output in the terminal for any compilation errors and fix them in your code.

By following these steps, you should be able to compile and run the C++ Bowling Score Calculator in VSCode successfully.

Happy coding!