WebAssembly を用いたブラウザ上で完結する C 言語の TDD 式学習システムの構築と提案
（コンパイラメッセージの日本語化と諸問題の解決について）

Construction and proposal of a TDD-style learning system for C that can be completed in a browser using WebAssembly
(Compiler messages translated to Japanese and resolution of various issues)

プロジェクトデザイン工学専攻（電気情報系）
小畠　一泰（指導教員　井上　浩孝）
Kazuhiro Kobatake

In recent years, programming education has become compulsory for elementary and junior high school students, and the importance of programming education has been increasing. Especially in the engineering field, C is one of the languages required to control microcomputers, but the form of teaching is still being explored. Technical colleges: KOSEN also have many opportunities for students to learn programming from the time they enter school, but problems such as "a large difference in students' proficiency level" and "students not understanding what they are creating" have occurred. These problems are thought to be caused by a combination of factors, such as "problems with the teaching and practice methods," "students copying their friends' code to complete assignments but not understanding them," and "students having few opportunities to learn outside of class time". In this research, we propose a system that can solve these problems without re-creating the lesson texts and can be easily integrated into the lessons. And, we have explored the Japanese localization of the compiler and examined countermeasures against problems caused by the LZ encoding. Since we have not yet been able to conduct a quantitative survey of the improvement in ability and measurement of the learning effect of Japaneseization by using this system, we would like to establish a measurement method and investigate in detail the extent to which the learning effect is manifested. In addition, when the system is used in classes, there is a good chance that the system will slow down or malfunction due to an instantaneous increase in the number of requests, so it will be necessary to conduct sufficient load tests to ensure that the system can withstand classroom use.

Keywords: *Programming Education, Improve Education, WebAssembly, JavaScript*
プログラミング教育，効率化，WebAssembly，JavaScript

１．はじめに

近年，小中学生のプログラミング教育が必修化されるようになり，プログラミング教育の重要性は高まっている．特に工学系ではマイコン制御などを行うために，C 言語が必要な言語の 1 つとなっているが，授業形態については未だ模索を重ねている段階だろう．

高専においても，入学時からプログラミングを学ぶ機会が多いが，「学生の習熟度に大きな差」や「自身が作っているものがわからない」といった問題が生じている．これらは「授業方法や実習方法に問題がある場合」や「友達のコードを写して課題をこなしているが，実際には理解していない場合」，「授業時間外で学習する機会が少ない場合」などの複合的な要因が原因であると考えられる．

本研究では，これらの問題を，授業資料を作り直すことなく解決し，授業に簡単に組み込むことができるシステムを構築，提案するものである．

２．本研究の目指すもの

本研究では，ブラウザ上で動作する軽量でテスト可能な学習環境を提案する．

2.1　ブラウザ上で動作する利点について

多くの場合において学生は自身のコンピューターに学校と同じ学習環境を構築するには一定以上のハードルがあるため，多くの学生が環境構築を諦めている傾向にあることが次よりわかる．

> 「自分の PC に予習や復習のために学校と同じ学習環境を整えたことはありますか」という問いに対する回答は“ない”が 35.3%，“ある，環

境構築は難しかった，面倒くさかった”が 29.4%，“あるが，環境構築は難しくて整えられなかった”と“ある，環境構築は簡単だった，面倒くさかった”，“ある，環境構築は難しかった，面倒くさくなかった”が 11.8%，“ある，環境構築は難しかった，面倒くさくなかった”が 0%と，自主学習への障壁の一つとなっていると考える．[1]より引用

したがって本システムであればブラウザを開くだけで学習を始めることができるため，学生のプログラミングを始めるハードルが低くなり自主学習を促すことへ繋がることが期待される．

2.1　従来の学習システムとの違いと優位性

呉工業高等専門学校 電気情報工学科の授業を例にすると Cygwin というソフトウェアを用いてプログラミング学習を行なっている．Cygwin とは“Windows OS 上に UNIX ライクな環境を提供する互換レイヤーで、フリーソフトウェアである[2]より引用”．したがって任意のテキストエディタで C 言語を記述後，Cywgin 上で動作する GCC(GNU Compiler Collection)によりコンパイル, 実行している．これによりコード記述には親しみがありGUI である Windows 上で行い，実行時のみ UNIX ライクな環境を用いることができる．

通常, プログラミング時には高機能な Visual Studio などの統合開発環境が用いられており，コード記述からコンパイル，実行，デバッグまで行うことができる．しかし高機能であるために満足に動作させるには必要スペックも高くなってしまうため演習室にあるパソコンでは十分に動かせないケースもあるだろう．これらの理由により開発環境を手軽に用意でき，UNIX ライクであるだけでなく統合された高速かつ軽量に動作する環境を提供するほうが好ましい．

３．Clang の日本語化について

本研究では C 言語のコンパイラやツールチェーンとして Clang を利用している．Clang は LLVM をフロントエンド，バックエンドとした C 言語のコンパイラであり，GCC(GNU Compiler Collection)の代替品を目標に開発されている．LLVM を利用しているためエディタと Clang との通信にて Language Server Protocol の提供や formatting などが可能となっている．本項では Clang におけるエラー文の日本語化について述べる．まず図１にあるプログラムを Clang でコンパイルすると図２のように未定義の変数を使用している旨を示すエラーが出る．これは図３のように Clang 内部で定義されているからだ．この時“%0”はプレースホルダーとなり，本エラーの場合変数名が入る．図３からわかるとおり Clang においてエラー内容はハードコーディングされているため国際化対応をする方法は存在せず，公式サイトにも次のとおり明記されている．

```c
#include <stdio.h>

int main() {
  printf("%d\n", aaaaa);
}
```

図１　コンパイルエラーの出るソースコード

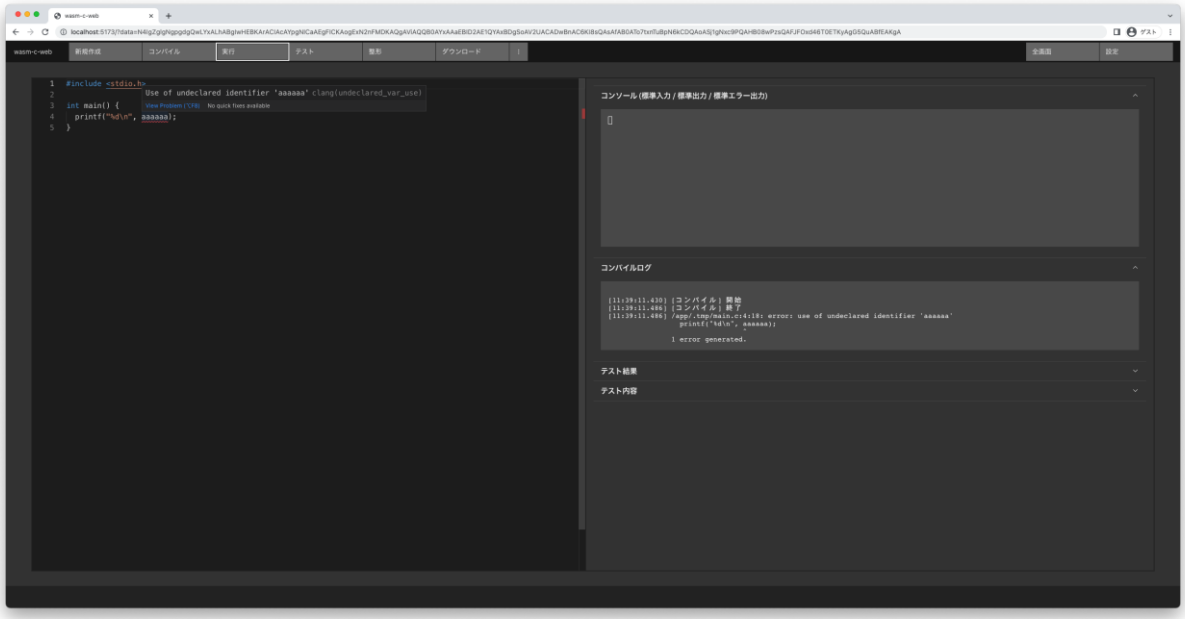

図２　通常のコンパイルエラーの様子

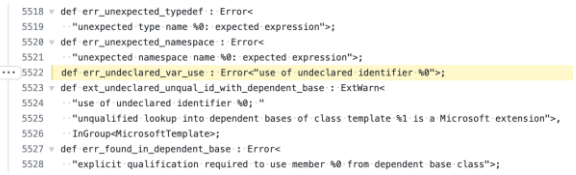

図３　Clang 内部のエラー文定義部

"Clang" CFE Internals Manual
The Clang "Basic" Library
The Diagnostics Subsystem
Adding Translations to Clang
Not possible yet! Diagnostic strings should be written in UTF-8, the client can translate to the relevant code page if needed. Each translation completely replaces the format string for the diagnostic. [3]より引用

したがって日本語化するには該当分を直接書き換える必要があり，GNU における sed コマンドを例にすると図４のとおりとなる．その結果を図５に，その時の様子を図６に示す．

```
$ sed -ie "s"/\
"use of undeclared identifier %0"/\
"定義されていない変数 %0 が使用されています/"\
"g"\
/path/to/Basic/DiagnosticSemaKinds.td
```

図４　エラー文定義部を書き換えるコマンド例

つまりすべてのエラーを順次日本語化すればエラー内容の日本語化が可能であるが，英語と日本語の

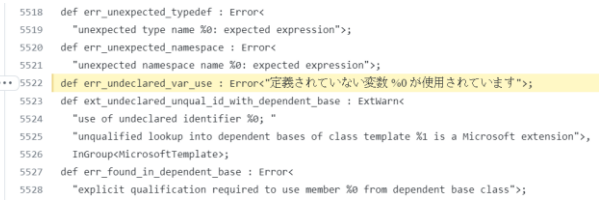

図５　エラー文定義部を書き換えた様子

翻訳において語順の解決が難しいため翻訳ツールを用いた機械的翻訳では変数の順序を保証できない．しかし Clang のエラー件数のうち 31%は未定義変数ということが[4]の調査結果からわかる通り、すべてのエラーを日本語化する必要はないことがわかる、このことから，全てのエラーを日本語化するという方針からエラーを順次通知し，報告数順に翻訳することで短期間で日本語化された Clang の提供が可能であることがわかる．現在このエラー報告システムの開発をおこなっている．

４．LZ 符号を用いた共有システムの課題

ソースコードやテストなど LZ 符号化し URL のパラメーターに埋め込むことで共有を手軽に行えるシステムだが，プログラムやテストの増加に比例して非常に長い URL になってしまうという問題点がある．例えば「N4IgZglgNgpgdgL（中略）IqJuZwgQEZjkQA（852 文字）」といった文字列があったときこれを復元すると図７のようにソースコードやテストが復元できる．しかし非常に文字列が長いため URL の共有方法は考える必要があるだろう．本研究ではソースコードやテストなどの塊をプロジェクトと呼び、図８のように一意のプロジェクト ID を付与しサーバー側で保管することとした．クライ

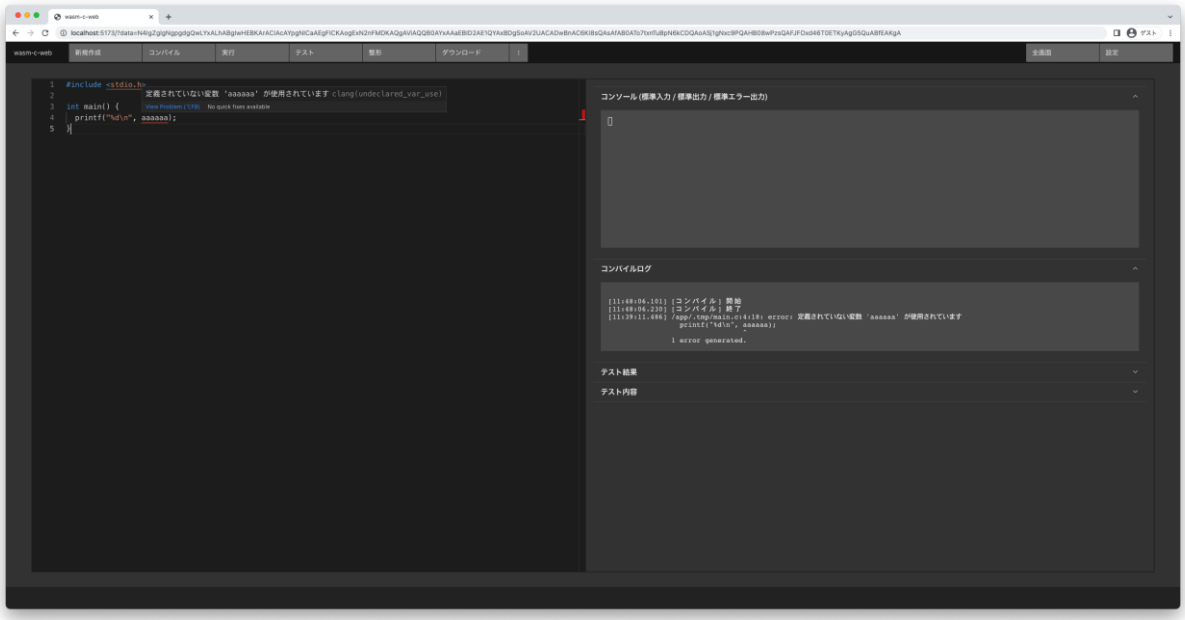

図６　コンパイルエラーが日本語化された様子

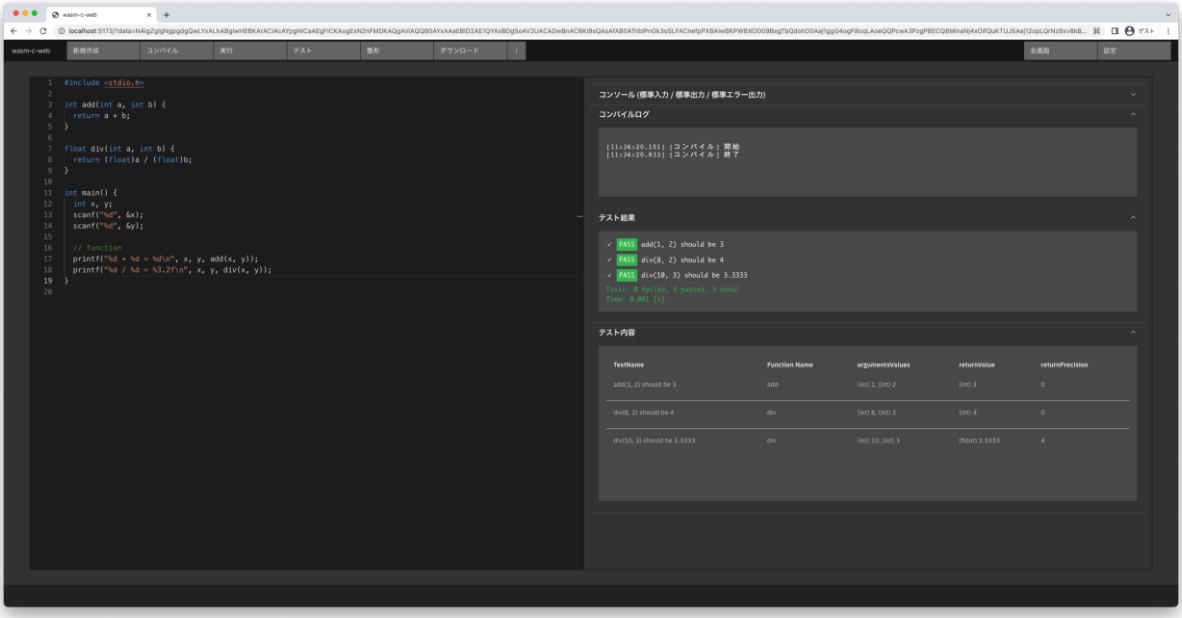

図７　LZ 符号化した文字列を復元した様子

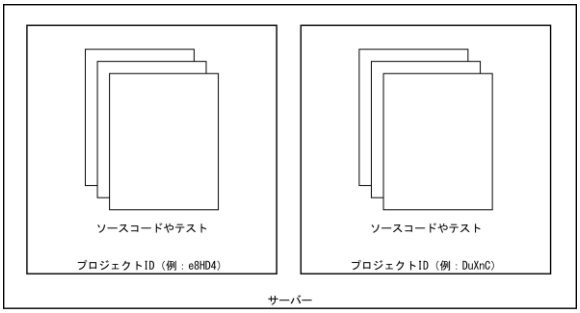

図５　エラー文定義部を書き換えた様子

アント側はプロジェクト ID がわかるとサーバー側に問い合わせ、各種ソースコードやテストを取得できる仕組みへと変更した．こうすることで今後実装する予定の複数ファイル化にも問題なく対応することができるだろう．

５．学習効果の測定方法について

先の研究では実際に利用した学生によるアンケートによって評価した定性的評価であり，教育効果を示すには従来の教育手法と比較して具体的にどう変化したか調べる必要がある．

効率化という側面で調査していく場合，従来のシステムと本システムにおいて補完機能によるタイプ数の変化やタイプ時間の変化，コンパイル，実行，テストをワンクリックで行えることによる総学習時間の変化などを計測しなければならない．

また，どの程度機能が利用されているかについてはクリック時に，時間やクリック場所などのイベント情報をサーバーに集約することで可能であろう．

加えて本来のタイプから補完機能によってどの程度効率化できたかについては Language Server Protocol の監視によって行え，入力と出力について文字数などを比較することで行えるだろう．

さらに学生ごとの学習度の習熟度合や授業時間外の利用に関しては，学生ごとにアカウントを発行し，利用してもらうことで継続的に計測可能だ．

６．まとめ

本研究ではコンパイラの日本語化についての模索と LZ 符号化によって生じた問題の対策方法の検討などを行った．現在，本システムを利用したことによる能力向上や日本語化による学習効果の測定について定量的調査を行えていないため，測定方法を確立させどの程度の学習効果が表れるのか詳しく調査していきたいと考える．また実際に授業にて利用する場合には瞬発的なリクエスト数の増加によるシステムの速度低下や不具合が起きる可能性が十分に考えられることから，負荷試験を十分に行い授業利用にも耐えられるシステムへとしていく必要があるだろう．

参考文献

[1] 小畠一泰，笠井聖二：WebAssembly を用いたブラウザ上で完結する C 言語の TDD 式学習システムの構築と提案，令和３年度　呉工業高等専門学校　電気情報工学科　卒業研究報告会予稿集，2022

[2] Cygwin - Wikipedia (https://ja.wikipedia.org/wiki/Cygwin), 2023.2.4 閲覧

[3] llvm-project/InternalsManual.rst GitHub (https://github.com/llvm/llvm-project/blob/main/clang/docs/InternalsManual.rst)，2023.2.4 閲覧

[4] Beaumont-GayMatt：Analyzing the Clang diagnostics experience. ，(https://llvm.org/devmtg/2012-11/Beaumont-Gay-Diagnostics.pdf)，2023.2.4 閲覧