

COBOL×AIハッカソン 2024 Day2

Geminiを活用した COBOLプログラム開発

チーム：CO-Geminiチャレンジャーズ

目次

1. COBOLプログラム開発で生成AIに期待すること
2. チャレンジ内容
3. Gemini活用シーン①：要件を提示しプログラムを生成する
4. Gemini活用シーン②：プログラムの改善点を指摘・改修する
5. Gemini活用シーン③：作成したプログラムの説明・プログラムへのコメント追加
6. Gemini活用シーン④：C0・C1カバレッジを満たすテストケースの生成
7. Gemini活用の所感

1. COBOLプログラム開発で生成AIに期待すること

- * COBOLプログラム開発を効率化したい
- * 下記を生成AIに期待する
 - ・プログラムの説明・理解
 - ・要件に応じたプログラムの生成
 - ・要件に応じたプログラムの修正
 - ・プログラム不良の修正
 - ・テストケースの生成

2. チャレンジ内容

* Geminiを活用して下記要件のプログラムを作成・テストする

■プログラム概要

2つの入力ファイルからデータを読み込み、年ごとに都道府県別の学校あたりの生徒数を計算してファイルに書き出す

■入力ファイル1：都道府県別学校数ファイル

レコード：年、都道府県、小学校、中学校、高等学校、大学・大学院

時点	地域コード	地域	小学校数【校】	中学校数【校】	高等学校数【校】
2021年	01000	北海道	984	579	272
2022年	01000	北海道	966	573	272
2023年	01000	北海道	950	563	270
2021年	13000	東京都	1328	801	429

■入力ファイル2：都道府県別生徒数ファイル

レコード：年、都道府県、小学生、中学生、高校生、大学・大学院生

時点	地域コード	地域	小学生【人】	中学生【人】	高校生【人】
2021年	01000	北海道	231714	122742	1153
2022年	01000	北海道	227372	120587	1121
2023年	01000	北海道	221397	119115	1092
2021年	13000	東京都	622820	311049	3017

■出力ファイル：都道府県別学校あたりの生徒数ファイル

レコード：年、都道府県、小学校あたりの平均生徒数、中学校あたりの平均生徒数、高校あたりの平均生徒数、大学・大学院あたりの平均生徒数

2. チャレンジ内容

- * プログラム開発の下記工程において、下記①～④のシーンでGeminiを活用する
- * マルチターンプロンプト（チャット）を活用

工程	Gemini活用シーン	ゴール
プログラムコーディング	①要件を提示し、プログラムを生成する	コンパイルが通るプログラムを作成する
	②プログラムの改善点の指摘・改修をする	指摘に従ってプログラムを改修する
	③作成したプログラムの説明・プログラムへのコメント追加	プログラムの理解ができる説明を生成し、プログラム中にコメントを追加する
プログラム単体テスト	④C0・C1カバレッジを満たすテストケースの生成	生成したテストケースを実行し、C0・C1カバレッジが100%となる

3. Gemini活用シーン①

要件を提示しプログラムを生成する

* 要件を追加・詳細化しながら繰り返しGeminiに問い合わせることで目的のプログラムを生成

■プロンプト①

添付の2つの入力ファイルから、年ごと、都道府県別の小学校、中学校、高等学校、大学・大学院の平均生徒数を算出し、ファイルに出力するCOBOLプログラムを作成して下さい。

ロジック誤り・コンパイルエラーに対して、
チャットでの会話で指摘し修正を繰り返し実施

⇒最終的なプログラム生成には下記を実施

- ・ 詳細な要件をまとめてプロンプトとする
- ・ Gemini回答を組み合わせてコーディング
- ・ 人手でのコンパイルエラーの解消

■プロンプト②（要件詳細化）

添付の2つの入力ファイルから、年ごと、都道府県別の小学校、中学校、高等学校、大学・大学院の平均生徒数を算出し、ファイルに出力するCOBOLプログラムを作成して下さい。

仕様を下記に示します。

- ・ 年と地域コードをキーに小学校、中学校、高等学校、大学・大学院の平均生徒数を算出する
- ・ school_all.csvを1行を読み込み、student_all.csvからキーが一致する行を探して平均生徒数を計算する。
- ・ ファイルの終端を判定するフラグはファイルごとに用意して管理してください。
- ・ 入力ファイルの1行目はヘッダ行として各カラム名が記述されているため読み飛ばす
- ・ 学校数・生徒数は整数8桁とする
- ・ 平均生徒数は整数5桁+小数2桁とする
- ・ 出力ファイルにはカラム名のヘッダを出力する
- ・ . . .
- ・ 各ファイルの仕様は後述の通り

■ファイル仕様

入力ファイル①：学校数 ファイル名：school_all.csv 形式：CSV
カラム：時点（年）,地域コード,地域,小学校数【校】,中学校数【校】,高等学校数【校】,大学・大学院数【校】,注記

入力ファイル②：生徒数 ファイル名：student_all.csv 形式：CSV

. . .

出力ファイル：平均生徒数 ファイル名：average_student.csv 形式：CSV

. . .

3. Gemini活用シーン① 要件を提示しプログラムを生成する

* コンパイルエラーの解消

コンパイルエラーメッセージの文面を与えるだけでは解消しきれない
⇒エラー原因可能性・具体的な対策案のインプットが必要

例) IF文ブロック、PERFORM文ブロックの中で終止符のピリオドが存在

■プロンプト

下記のコンパイルエラーを対策してください。
"PERFORM"文に対応する"END-PERFORM"がありません。
"END-PERFORM"に対応する"PERFORM"文がありません。

■Gemini回答

PERFORM 文の終わりにEND-PERFORM を追加し関連付けました。

```
PERFORM READ-SCHOOL-RECORD.  
PERFORM UNTIL WS-END-OF-SCHOOL-FILE = 'Y'  
  PERFORM READ-STUDENT-RECORD.  
  IF WS-YEAR = WS-YEAR-2 AND WS-REGION-COD  
    PERFORM CALCULATE-AVERAGES  
  END-IF.  
  PERFORM READ-SCHOOL-RECORD.  
END-PERFORM.
```

* ロジック誤りの対処

例) 入力ファイル1行分のみしか処理されない⇒ファイル終端判定フラグに問題

■プロンプト

ファイル終端判定フラグ管理が2つのファイルで重複しているため、修正してください。

■Gemini回答

・・・これは、2つのファイルの処理を同期させるために、片方のファイルの EOF で両方の処理を終了させる意図があると考えられますが、より明確な方法があります。

修正案

ファイル終端判定フラグを別々に管理し、SCHOOL-FILE と STUDENT-FILE それぞれに対応する EOF フラグを作成します。

```
...  
READ SCHOOL-FILE  
AT END MOVE 'Y'  
  TO WS-END-OF-FILE.  
...  
READ STUDENT-FILE  
AT END MOVE 'Y'  
  TO WS-END-OF-FILE.  
...
```

元処理の意図を推測
修正の妥当性判断の助けになる

4. Gemini活用シーン② プログラムの改善点を指摘・改修する

- * 下記プロンプトでエラーハンドリングの不足箇所の指摘・エラー処理の追加

■プロンプト

<プログラムの仕様説明>

上記プログラムで想定されるエラーケースを教えてください。

下記の2つのエラーケースについて、エラー対応策に従ってソースを修正して下さい。

- ・キー一致エラー
- ・ゼロ除算エラー

- * 改善観点を提案してくれるため、レビュー・リファクタリングに有効
- * 人手での視点と組み合わせることで観点漏れの防止になる

■Gemini回答

1. ファイル関連
...
2. データ関連
...
 - ・ キー一致エラー: school_all.csv に存在するキー（年と地域コード）が student_all.csv に存在しない場合、平均生徒数を計算することができません。
 - ・ ゼロ除算エラー: 学校数が0の場合、平均生徒数を計算する際にゼロ除算が発生し、プログラムが異常終了する可能性があります。
3. ロジック関連
...
4. その他
...
エラー対応策
 - ・ キー一致エラー: どちらかのファイルにキーが存在しない場合は、エラー処理を追加してプログラムを終了するか、または適切なメッセージを出力する必要があります。
 - ・ ゼロ除算エラー: 学校数が0の場合、平均生徒数を計算せず、エラーメッセージを出力する必要があります。
 - ...

5. Gemini活用シーン③

作成したプログラムの説明・プログラムへのコメント追加

* 下記プロンプトでプログラムの説明およびコメント追加を試みた

■プロンプト

添付のcobolプログラムを解説してください。

このプログラムの入力ファイルとして例えば添付CSVファイルを使用します。これを踏まえて、プログラムの処理内容を詳しく説明してください。

詳しい説明をプログラム中にコメントとして記載してください。

* プログラムの用途や動作例の解説もあり、理解の助けになった

■ Gemini回答

プログラムの用途

このプログラムは、学校と生徒に関するデータから、地域別の平均生徒数を計算し、CSVファイルに出力するプログラムです。このプログラムは、教育機関のデータ分析や、地域別の生徒数の推移を把握する際に役立ちます。

プログラムの動作例

学校ファイル "school_all.csv" に以下のデータが含まれているとします。

.....

このプログラムを実行すると、"average_student.csv" に以下のデータが出力されます。

.....

■ Geminiがコメント付与したプログラム

```
1 IDENTIFICATION DIVISION.  
2 PROGRAM-ID. AVERAGE-STUDENT.  
3  
4 *> このプログラムは、学校データと生徒データから地域コード・年別に平均生徒数を計算し、出力するプログラムです。  
5 *> 入力ファイル: school_all.csv (学校データ)  
6 *> 入力ファイル: student_all.csv (生徒データ)  
7 *> 出力ファイル: average_student.csv (平均生徒数データ)  
8  
9 ENVIRONMENT DIVISION.  
10 INPUT-OUTPUT SECTION.  
11 > FILE-CONTROL.  
22 DATA DIVISION.  
23 FILE SECTION.  
24 FD SCHOOL-FILE.  
25 *> 学校ファイルの1レコードを格納するデータ項目  
26 01 SCHOOL-RECORD PIC X(255).  
27  
28 FD STUDENT-FILE.  
29 *> 生徒ファイルの1レコードを格納するデータ項目  
30 01 STUDENT-RECORD PIC X(255).  
31  
32 FD AVERAGE-FILE.  
33 *> 平均生徒数データファイルの1レコードを格納するデータ項目  
34 01 AVERAGE-RECORD PIC X(255).  
35  
36 WORKING-STORAGE SECTION.  
37 01 WS-SCHOOL-DATA.  
38 05 WS-YEAR PIC 9(4)X(2). *> 年  
39 05 WS-REGION-CODE PIC X(5). *> 地域コード  
40 05 WS-REGION PIC X(20). *> 地域名  
41 05 WS-ELEMENTARY-SCHOOL-COUNT PIC 9(8). *> 小学校数  
42 05 WS-JUNIOR-HIGH-SCHOOL-COUNT PIC 9(8). *> 中学校数  
43 05 WS-HIGH-SCHOOL-COUNT PIC 9(8). *> 高校数  
44 05 WS-UNIVERSITY-COUNT PIC 9(8). *> 大学数  
45 05 WS-NOTE PIC X(20). *> 注記  
46  
47 01 WS-STUDENT-DATA.  
48 05 WS-STUDENT-YEAR PIC 9(4)X(2). *> 年  
49 05 WS-STUDENT-REGION-CODE PIC X(5). *> 地域コード  
50 05 WS-STUDENT-REGION PIC X(20). *> 地域名
```

6. Gemini活用シーン④

C0・C1カバレッジを満たすテストケースの生成

- * 下記プロンプトでテストケースおよびテストデータ・実行結果を提示させてみた

■プロンプト

このプログラムのC0カバレッジおよびC1カバレッジが100%となるテストケースとその入力データを作成してください。
各テストケースの実行結果も提示してください。

- * Geminiが生成したテストデータ・出力結果の誤りもあった

⇒テストケースとテストデータの内容があっているかは人手でも確認が必要

■ Gemini回答

ケース1: 正常系 - 1件の学校と生徒データが存在する場合

school_all.csv

時点(年),地域コード,地域,小学校数【校】,...

2023,00001,東京,100,50,30,20,東京都

2023,00002,大阪,80,40,25,15,大阪府

student_all.csv

...

ケース2: 正常系 - 複数の学校と生徒データが存在する場合

ケース3: 正常系 - 年と地域コードが一致するデータがない場合

ケース4: 正常系 - 学校数または生徒数が0の場合

ケース5: 異常系 - school_all.csvにデータが存在しない場合

ケース6: 異常系 - student_all.csvにデータが存在しない場合

ケース7: 異常系 - school_all.csvのデータが不正なフォーマットの場合

ケース8: 異常系 - student_all.csvのデータが不正なフォーマットの場合

6. Gemini活用シーン④

C0・C1カバレッジを満たすテストケースの生成

■ 日立COBOL(COBOL2002)のカバレッジ機能(※)と連携したC0・C1カバレッジの網羅

- * カバレッジ機能で出力したプログラム各行のC0・C1カバレッジ網羅状況の情報を与えることで Geminiが未実行の行を抽出し必要なテストケースを提示してくれる

■ プロンプト

csv出力

日立COBOL カバレッジ機能

<ソースカバレッジ情報ファイル>

"C0"	"C1"	行番号	ソーステキスト
"	"	000097	PROCEDURE DIVISION."
"*	"	000098	OPEN INPUT SCHOOL-FILE STUDENT-FILE"
"	"	000099	OUTPUT AVERAGE-FILE."
"	"	000100	"
"*	"	000101	WRITE AVERAGE-RECORD FROM WS-COLUMN-NAME."
"	"	000102	"
"*	"	000103	READ SCHOOL-FILE"
"	"	000104	AT END MOVE "Y" TO WS-EOF-FLAG-SCHOOL."
"	"	000105	"
"	"@"	"	"
"*	"	000106	READ STUDENT-FILE"
"	"	000107	AT END MOVE "Y" TO WS-EOF-FLAG-STUDENT."
"	"	000108	"
"	"@"	"	"
"*	"	000109	PERFORM READ-SCHOOL-FILE."
"*	"	000110	PERFORM READ-STUDENT-FILE."
"	"	000111	"
"*	"@"	000112	PERFORM PROCESS-DATA UNTIL WS-EOF-FLAG-SCHOOL."
"	"@"	000113	"

添付のソースカバレッジ情報ファイルは、C0カバレッジ、C1カバレッジの網羅状況を示しています。

<ソースカバレッジ情報ファイルの仕様説明>

C0カバレッジ、C1カバレッジが未実行・未通過の行を示してください。

■ Gemini回答

■ C0, C1カバレッジ未実行・未通過行

下記の行がC0, C1カバレッジで未実行・未通過となっています。

行番号	ソーステキスト	C0	C1
104	AT END MOVE "Y" TO WS-EOF-FLAG-SCHOOL.	.	.
107	AT END MOVE "Y" TO WS-EOF-FLAG-STUDENT.	.	.

■ テストケースと入力ファイル

テストケース5: school_all.csvファイル終端処理

目的: school_all.csvファイルの終端処理で、AT END句が実行されることを確認します。

テストケース6: student_all.csvファイル終端処理

目的: student_all.csvファイルの終端処理で、AT END句が実行されることを確認します。

⇒テストとカバレッジ情報のGemini連携の繰り返して
C0・C1カバレッジ100%を実現できる

※カバレッジ機能

プログラム実行をトレースしてテスト実施後のC0カバレッジ、C1カバレッジ、未実行情報などのカバレッジ情報を表示できる機能 (csv形式またはテキスト形式)

7. Gemini活用の所感

* 利点

- 曖昧な要件からもたたき台となる成果物を生成可能
⇒説明を付与してくれるため、調査・確認のきっかけとなるキーワードが得られる
- 広く観点を提案してくれるため、人による作業と組み合わせることで観点漏れが防止できる
- Geminiとの対話を通じて理解が進み、要件の詳細化ができる
⇒相談相手として活用可能
- カバレッジ機能と連携することでC0・C1カバレッジ網羅が可能

* 工夫が必要な点

- 要件を詳細化する
指示の具体化、ファイルや表の形式を明示する、など
- 複数パターンのプロンプトでの結果を組み合わせ、適切な結果を取捨選択する
- プログラムやデータなどの成果物の生成結果に対しては妥当性のチェックが必要