

バイブコーディングで広がる！ 生成AI×Processing授業



東京学芸大学附属高等学校

情報科 飯田秀延

バイブコーディングとは

生成AIと人間が「雰囲気」や「ノリ」で対話しながら、自然言語での指示に基づいてソフトウェアを開発する新しい手法

「もう構文のことなんか考えなくていい。ただやりたいことをAIに伝えるだけで動く」



Andrej Karpathy ✓

@karpathy · フォローする



There's a new kind of coding I call "vibe coding", where you fully give in to the vibes, embrace exponentials, and forget that the code even exists. It's possible because the LLMs (e.g. Cursor Composer w Sonnet) are getting too good. Also I just talk to Composer with SuperWhisper [さらに表示](#)

午前8:17 · 2025年2月3日



3万



返信



リンクをコピー

1,346件の返信を読む

「プログラミング」の授業はこの辺り(年間で10時間前後)

Contents

1章 Theory

情報社会

1	情報とその特性	2
2	メディアとその特性	4
3	問題を解決する方法	6
4	情報の収集と分析	8
5	解決方法の考案	10
6	知的財産	12
7	個人情報	14
8	情報セキュリティ	16
9	情報モラルと個人の責任	18
10	情報技術の進歩と役割	20
11	情報技術が社会に与える光と影	22
	章末資料	24
1	検索の順位付け	
2	SNSへの投稿	
3	サイバー犯罪	
4	ICTの普及と識字の関係	
5	人工知能と人間の仕事	
6	インターネットと選挙	
	私の問題解決	27
	1章のまとめ	28
	章末問題	29

2章 Theory

情報デザイン

12	コミュニケーションとメディア	32
13	情報のデジタル化	34
14	数値の表現	36
15	2進法の計算	38
16	文字のデジタル表現	40
17	音のデジタル表現	42
18	画像のデジタル表現	44
19	データの圧縮	46
20	デジタルデータの特徴	48
21	メディアと文化の発展	50
22	ネットコミュニケーションの特徴	52
23	情報デザイン	54
24	操作性の向上と情報技術	56
25	全ての人に伝わるデザイン	58
26	コンテンツ設計	60
	章末資料	62
1	機種依存文字	
2	色の表現	
3	ラスタ形式とベクトル形式	
4	ピクトグラムと東京オリンピック	
5	デザインで問題解決	
6	色の識別性	
	私の問題解決	65
	2章のまとめ	66
	章末問題	67

3章 Theory

プログラミング

27	コンピュータの構成	70
28	ソフトウェア	72
29	処理の仕組み	74
30	論理回路	76
	アルゴリズムの効率性	80
	プログラムの仕組み	82
	プログラミング入門	84
	プログラムの応用	86
	問題のモデル化	88
	デジタルの活用	90
38	シミュレーション	92
39	シミュレーションの活用	94
	章末資料	96
1	補助記憶装置	
2	時間を主役とした地図	
3	固定小数点表現と浮動小数点表現	
4	プログラミング言語の年表	
	私の問題解決	99
	3章のまとめ	100
	章末問題	101

4章 Theory

ネットワークの活用

40	情報通信ネットワーク	104
41	デジタル通信の仕組み	106
42	インターネットの利用	108
43	安全安心を守る仕組み	110
44	情報システム	112
45	さまざまな情報システム	114
46	情報システムの信頼性	116
47	データの活用とデータベース	118
48	データの管理	120
49	データの収集と種類	122
50	データの分析	124
51	不確実な事象の解釈	126
52	2つのデータの関係	128
	章末資料	130
1	通信規格の推移	
2	海底ケーブル	
3	ルートDNSサーバ	
4	生体認証	
5	グラフを疑う	
6	ギャンブラーの誤謬	
	私の問題解決	133
	4章のまとめ	134
	章末問題	135

5章 Action

問題解決

53	検索のコツ	138
54	仕事の研究	139
55	アイデアの大量生産	140
56	光の三原色体験	141
57	データ量の見積もり	142
58	図解表現	143
59	ピクトグラム	144
60	部活紹介CM	145
61	Webニュースページ	146
62	作図しよう	148
63	プログラムで動きを再現	149
64	お知らせセンサー	150
65	気まぐれAI	152
66	左右限定あっち向いてホイ	154
67	プログラムの改善	156
68	Myお天気キャスター	158
69	災害時帰宅マップ	160
70	シミュレーションをしよう	161
71	誕生日シミュレーション	162
72	高校生の実態調査	163
73	コンビニデータベース	164
	章末資料	166
1	意思伝達装置を活用した物理学者	
2	SDGs	
3	プレゼンテーション	

本校の場合

Python 6時間
Processing 4時間

で実施



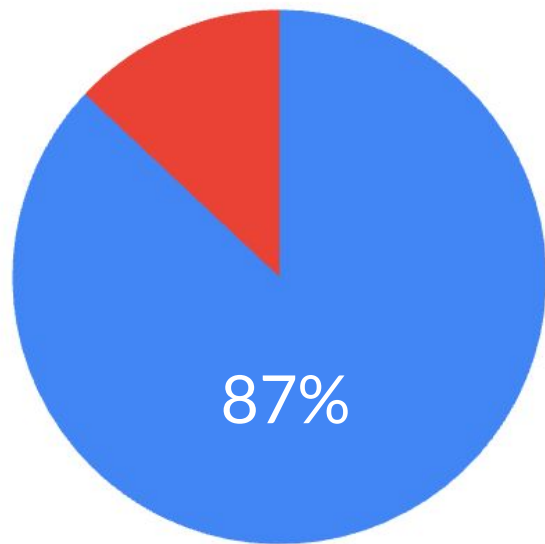
Processingは「楽しい」
しかし・・・

- ・新たな言語習得は大きな負担
- ・プログラミングスキルがないと「やりたいこと」が実現できない

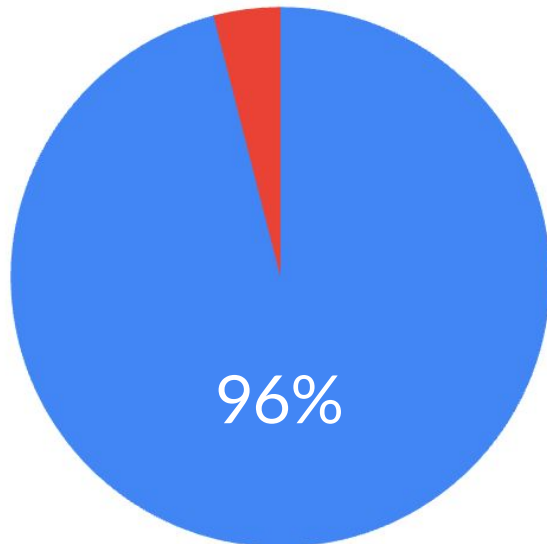
生成AIで補助
(バイブコーディング)

新たなプログラミング言語を習得する負担を軽減しつつ、「やりたいこと」を実現するためのスキルを補うことによって、プログラミング教育の学習効果を高めることが期待できる。

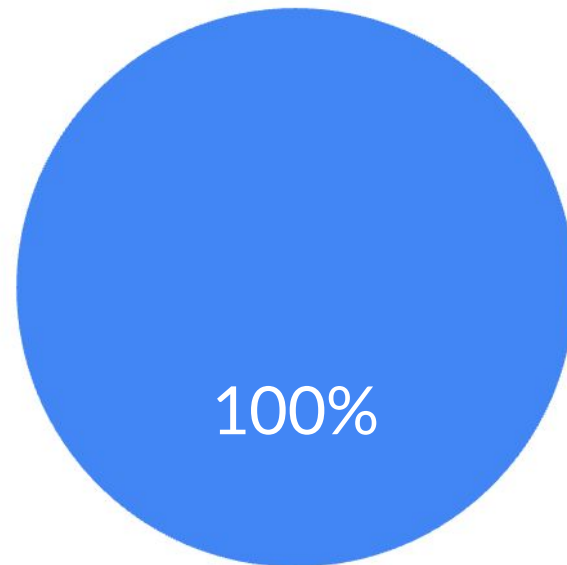
生成AIを使用した生徒の割合



一昨年



昨年



今年

- ・生成AIの性能の向上
- ・探究の授業で全員がNotebookLMを活用

制作時間:3時間

評価方法:ループリックに
基づいた相互評価

禁止事項:公序良俗に反するもの
他人を傷つけるもの

その他 :作ったものをSNSなどに
安易に公開しないこと

Processing実習 ルーブリック

項目	評価				
	1	2	3	4	5
技術 (4)	短時間で適当に作った感じ。工夫もない。 真面目に取り組んでいない	作品はできているが、生成AIからの出力をほぼそのまま使っている	プロンプトを何度か手直しして、自分なりの作品を完成させている	生成AIの出力に加えて独自の工夫を行い、他人に真似のできない作品ができている	生成AIとのやりとりを何度も重ね、独自の工夫が複数入っている、他人に真似のできない作品ができている。
努力 (3)	あまり努力の様子が見られない 真面目に取り組んでいない	課題にそれなりに取り組んでいる	授業時間に真面目に取り組んだ様子が見とれる	授業時間に集中して真面目に取り組んでいた様子が見とれる	授業時間に非常に集中して、意欲的に新しいことにチャレンジしている
独創性 (3)	独創性が感じられない ありきたり	やや工夫の跡が見受けられる	工夫の跡は見られるが、すぐに思いつきそうなもの	オリジナリティが感じられる	オリジナリティがある上、誰にも真似ができないようなアイデアがある

この授業で期待すること

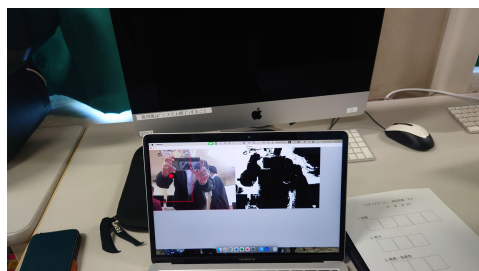
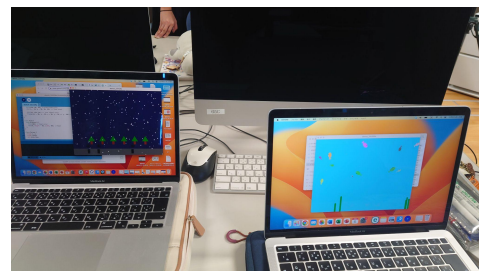
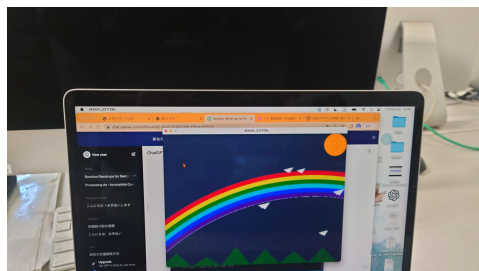
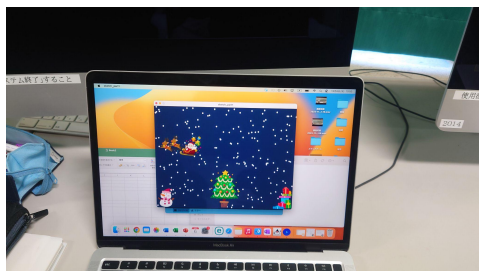
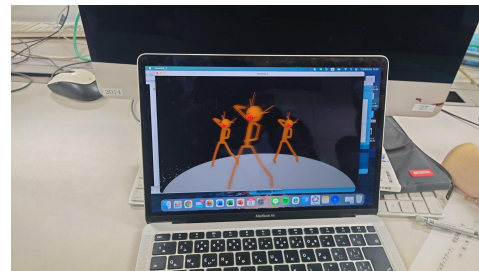
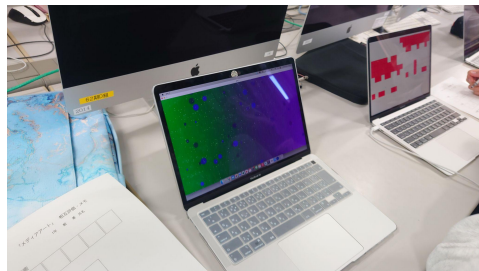
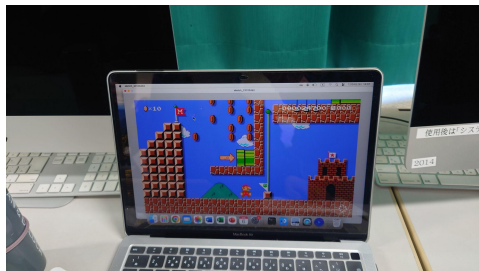
新しいプログラミング手法の習得

高度な知識と技術の取得

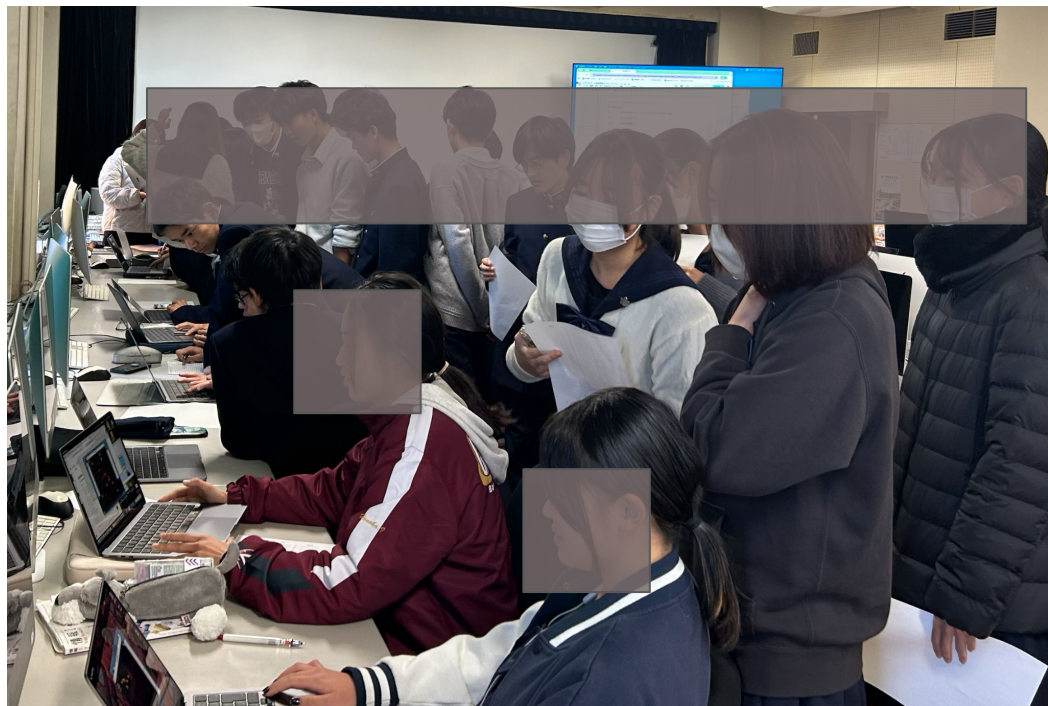
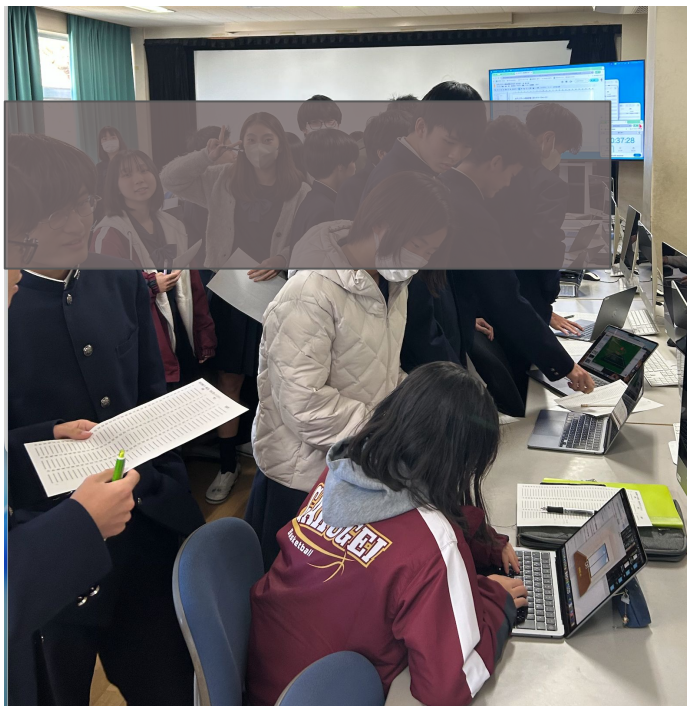
独創的なアイデアの発露

生成AIの特徴の理解

作品例の紹介(実演)



相互評価(ギャラリーウォーク)の様子



今回の生成AIを活用したプログラミング(バイブコーディング)は、どうでしたか(一部紹介)

自分でコードを一から考える必要が無いので、**とても楽だった**。せっかく良いアイデアが浮かんでも自分1人では形にすることができない。しかし、AIを使うと自分の指示通りに作ってくれるので、やりたいことが**すぐにできた**。

エラーが出た時にどの行か教えて何が原因かを教えてもらい次に活かした。**プロンプトの主語や目的語を明確にした**。

生成AIは作るのに簡単だったけれど、**自分のやりたいことを完璧に伝えるのがとても難しかった**。

生成AIを使った今回のプログラミングは、**思いついたアイデアをすぐ形にできるところが便利**だった。自分だけだと難しい部分も、AIに質問して調整できて、試行錯誤の速度が大きく上がりました。また、コードを書くよりも**「こうしたい」という発想をたくさん出せて、独創性が現れるゲームを作れた**。

生成AIに望んだことを言うだけでいい感じに作ってくれるので、**時代ってすごい**なと思いました。やりたいように好きなようなゲームがお願いするだけでできていく、自分の今までのゲームの好きな傾向がわかりました。しかし、意外と単純のゲームの方が面白かったり、面白いの難しさも考えました。

自分なりに工夫した点や苦勞した点を教えてください(一部紹介)

大規模な変更ではなく、**機能追加ごとに指示を分け、コードの一貫性を保つようにした**。特に、何かを付け加えたいと思った際に、ただ加えるだけではうまく行かず、時には大幅な修正を行わなければならないときもあったため、**プログラミング技術の難しさを改めて思い知らされた**。

生成AIにわかりやすく、優しく伝えるようにした。苦勞した点は、図を使って説明できないので、**プロンプトの内容を考えることに苦勞した**。

AIが全肯定botにならないように、**プロンプトを工夫して厳しくみってもらうようにした**。ゲームの仕様についての改善点は自分でプレイして確認、処理を軽くするか技術的な面はAIに確認してもらうなど役割を分担した。

processingを使うのが初めてだから、全部 AIに作って貰えばいいと最初は思っていたけど、**的確に、段階的に指示をしないとコードが変わったり、エラーが大量に出てきたりする**からそうはいかなかった。ジャンピングゲームの中にストーリーや設定を丁寧にふくめた。

生成AIが前の要望を反映させてくれないことが多かった。だからイタチごっこみたいになって大変だった。また、**エラーも出ることが多く、直すのも難しかった**。自分で工夫したのはサンプルの盤面をリセットするようにしたこととヒントを出すようにしたことだ。

今後、生成AIをどのように活用していきたいと思いますか(一部紹介)

思考判断を委ねきっちゃったらそれは良くないと思うけど、作業をする上でうまく命令して使っていければ自分のレベルを上げることにもつながると思うし、自由な時間を作れる。適切な使い方をしていきたい

Aiに完全頼り切るのではなくAiが得意とすること(計算や作業)はAiに任せて創造的な判断や設計の質を高めることは自分でしていきたい。

使いすぎても、あまりいいことがないんだなあと言ったイメージ。全部頼んでしまうようなことをすると、自分が何をしたいのかわからなかったりするんで、アイデアをもらうくらいの頻度で使うのが適切なのだと思います。

今後いろんな場面で使う場面が増えると思うけれど、その時も気をつけながら使っていきたいなと思った。

エラーが自分で理解できるくらいプログラミングを学んでから生成AIを使うことで、もしミスがあってもすぐ修正できるようになるのではないかと思った。

今回のvibe codingでは自分がやりたいことはたくさんあるのに、それが思うように実行されないもどかしさから、自分もプログラミングに関する十分な知識を持った上で、AIが提示したものを補えるような形で活用したいと思った。

自分でテーマを設定し、AIが長いコードを作成するといった、役割にあった活用していきたい。AIに頼りすぎることなく、倫理に逸れない使い方を社会に出ても続けられるように習慣化させることが大切だと感じた。

プログラミングの授業に生成AIを活用すると・・・



新たなプログラミング言語を習得する負担を軽減しつつ、「やりたいこと」を実現するためのスキルを補うことによって、プログラミング教育の学習効果を高めることが期待できる。

※ただし、基本的なコーディングの知識は必要

生成AIに依存し過ぎると人間はバカになるのか？実験が明らかにした、生成AIを賢く使える人、使えない人

6/8(土) 17:21 配信  39   



■ 苦手なタスクで生成AIを使うリスク

米テキサス州にあるアビリーンクリスチャン大学の研究者らが発表した論文によれば、彼らは21人の大学生を被験者として、一定の条件下でコーディングをさせた。その21人は、コーディングの授業を受講していた学生たち、つまりコーディングの初心者だったとのことだ。

そんな被験者に対して、C++というプログラミング言語を使い、「ループを使って正の数と負の数のどちらが多いかを判定するプログラム」を作成するというタスクを与え、さらにChatGPTとGitHub Copilot（コード生成に特化した生成AIサービス）の2種類のツールを使える環境にした上で、実際の開発を行わせた。

ちなみに「ループを使う」という具体的な指示が行われた理由は、ループの概念が実験の2週間前に授業で示されていた、つまり「被験者にとって、ループはまだ完全に習得されたとは言えないが、圧倒的に難しいというほどではない概念だった」ためだった。

えたのかを確認したのである。

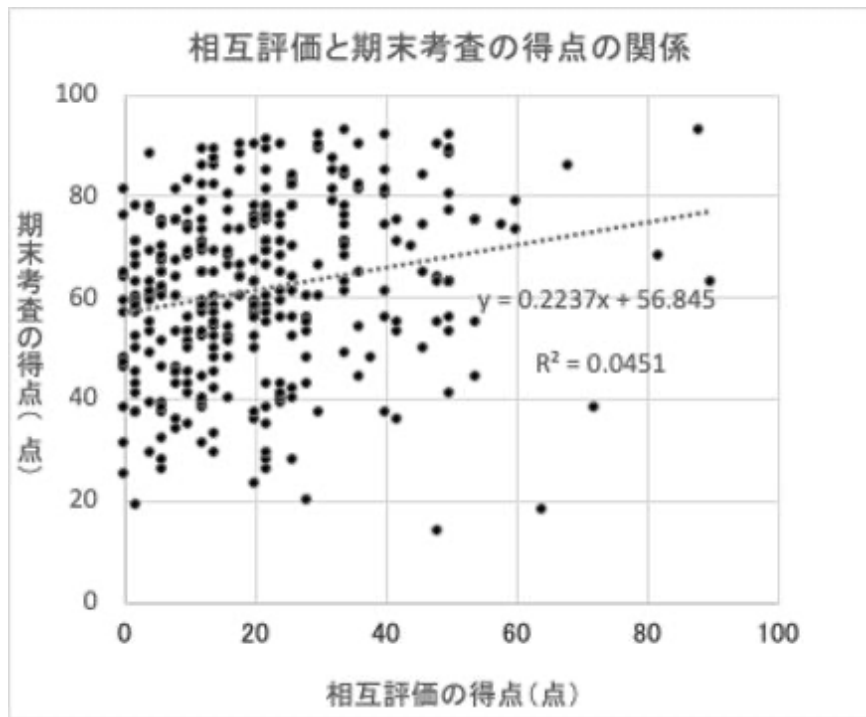
結果はどうだったか。結論から言うと、もともとコーディングが得意な学生については、AIの提案を適切に取捨選択しながら効率的に課題を解くことができた一方で、コーディングが苦手な学生は、AIの提案に振り回されて本質的な理解が追いつかず、結果的に自分の力でプログラムが書けなくなってしまう傾向が見られたそうである。

もともとコーディングの授業において良い成績を収めていた学生は、コード生成AIの提案を吟味し、適切なものを選択した。その結果、効率的に課題に取り組むことができ、自分の考えを優先しながらAIを活用する姿勢が見られた。そうした学生は、AIを使いながらも自分の知識を深めることができたと研究者らは結論付けた。

「プログラミングが得意な生徒ほど生成AIをうまく使いこなせる」は本当か？

うだ。

※一昨年の結果(昨年もほぼ同じ)



相互評価(生成AIをうまく使いこなせていたか)と期末考査の得点の相関係数は0.21

相関係数の目安

0.0~0.2 ほとんど相関関係がない

0.2~0.4 やや相関関係がある

0.4~0.7 かなり相関関係がある

0.7~1.0 強い相関関係がある

「プログラミングが得意であることと生成AIをうまく使いこなせることはあまり関係がない」？！

生成AIによる評価の難しさの要因5つ

1. 学習アプローチの違い

学習アプローチやスタイルが異なるため、一部の生徒は伝統的な教科の成績が高くても、生成AIのような新しい分野に適応するのが苦手なのかもしれない。

2. 興味や動機の違い

生徒が興味を持っている分野や、将来の職業への動機が生成AIとは無関係であるのかもしれない。

3. 成績評価の偏り

学校の成績評価体系が、生成AIでのスキルや知識を正確に測定できないのかもしれない。

4. 教育カリキュラムの不足

学校の教育カリキュラムがまだ生成AIや先端技術に焦点を当てていないのかもしれない。

5. 個々の適性や才能

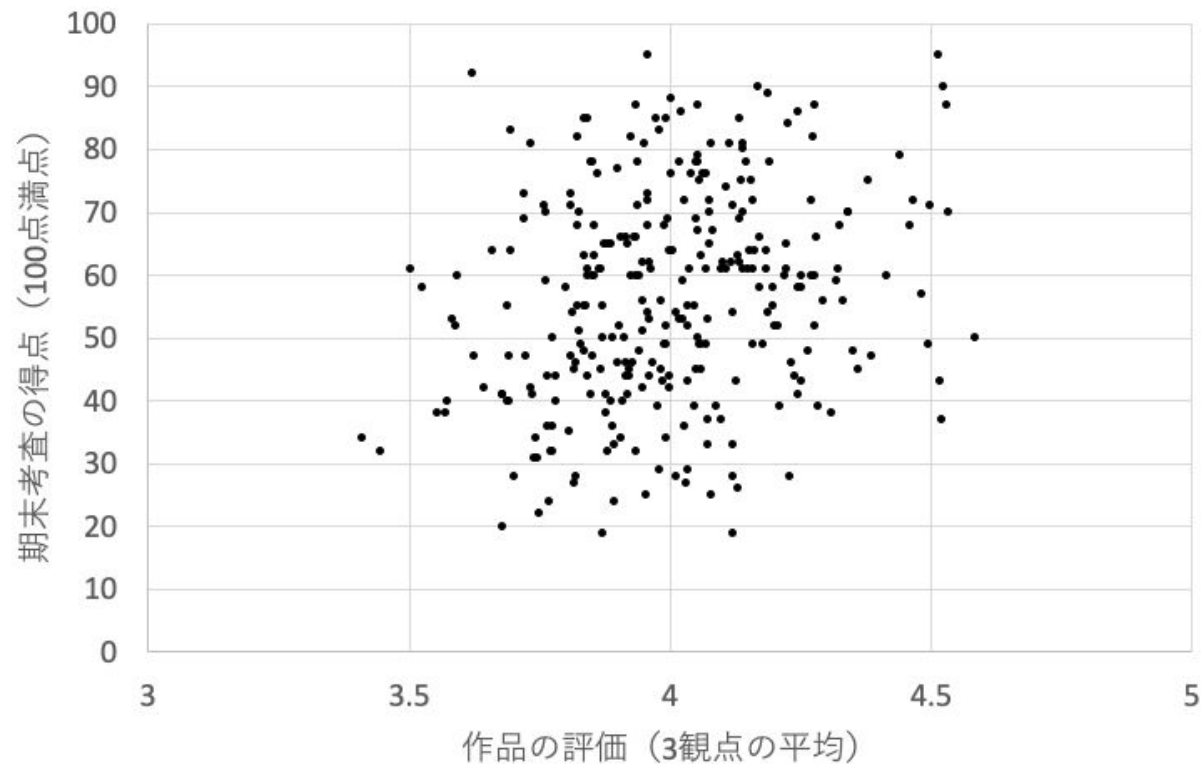
生成AIを使いこなすための適性や才能が、他の教科の成績と直接関連しないのかもしれない。

→今回の評価規準(基準)がよくなかった



※今年の結果(作品の評価を改善)

作品の評価と期末考査の得点の関係



相関係数:0.25

3年連続で相関係数は0.2前後
もはやこう言い切って良いのでは？

「プログラミングが得意であることと生成AIをうまく使いこなせることはあまり関係がない」

「評価」をどうするか？

作品(アウトプット)を評価することはできるが、
「生成AIをうまく活用できたこと」(過程)をどのように評価
するのか？

「生成AIをうまく使いこなせること」の適切な
評価規準とは何なのか？



「生成AIをうまく使いこなせること」の適切な評価規準とは何なのか？



そもそも「生成AIをうまく使いこなせること」を評価する必要ってあるのか？

「電卓をうまく使っているか」の評価などしない



バイブコーディングを3年続けた感想

基本的なコーディングの知識を取得している生徒が
バイブコーディングで作品を制作すると
自分の考えを優先しながらAIを活用することや
AIを使いながらも自分の知識を深めることができ、
プログラミング教育の学習効果を高めることが期待できる。



「評価」をどのようにするのが今後の課題

基本的なコーディングの知識を取得している生徒が
バイブコーディングで作品を制作すると
自分の考えを優先しながらAIを活用することや
AIを使いながらも自分の知識を深めることができ、
プログラミング教育の学習効果を高めることが期待できる。

「評価」をどのようにするのが今後の課題



次に試してみたいこと

「生成AIでソースコードを評価する」