




The Robots Are Coming: Exploring the Implications of OpenAI Codex on Introductory Programming

Authors: James Finnie-Ansley, Paul Denny, Brett A. Becker, Andrew Luxton-Reilly, James Prather



What is Codex

- Codex is a model developed by OpenAI, descendant of GPT-3
- Trained on more than 50 million GitHub repositories. It powers GitHub Copilot.
- Can interpret simple commands in natural language and generate working code.
- Most capable in Python, also proficient in dozen languages including JavaScript, Go, Swift, and Shell.

More about..

- Codex
 - “Davinci” Codex model (the most capable, but slowest)
 - A temperature of 0.9
 - Higher temperature values produce more random responses
- The Rainfall Problem is a classic problem used in computing education to assess programming ability, particularly in introductory courses.

Research Questions

RQ1: How does Codex perform on first year assessments compared with CS1 students?

RQ2: How does Codex perform on variations of a benchmark computing education problem that differ in context and level of detail?

RQ3: How much variety is there in the solutions generated by Codex?

Test Structure and Evaluation Criteria

- Using 23 programming questions appeared on 2 CS1 programming tests. 11 question from test 1 and 12 questions from test 2.
- Test 1 with 20 marks, with 2 questions worth 1 mark and other worth 2 marks.
- Test 2 is with 25 marks with 2 questions worth 1 mark, 3 questions worth 3 marks and other worth 2 marks.
- Each question included the problem statement and example test cases

Write a function `date_string(day_num, month_name, year_num)` that returns a string in the format "day month, year".

See the examples below for more information.

For example:

Test	Result
<code>print(date_string(1, "December", 1984))</code>	1 December, 1984
<code>print(date_string(1, "March", 1984))</code>	1 March, 1984

Reset answer

```
1 def date_string(day_num, month_name, year_num):
2
3
4
5 """
6 Write a function date_string(day_num, month_name,
7 year_num) that returns a string in the format
8 "day month, year".
9 >>> print(date_string(1, "December", 1984))
10 1 December, 1984
11 >>> print(date_string(1, "March", 1984))
12 1 March, 1984
13 """
```

Test Structure and Evaluation Criteria

For students

- Exam conditions, time limits and invigilation
- Lab-based software for automatic grading
- Immediate feedback, displaying fail test cases after each submission
- Penalize incorrect submission by 5%, up to 50%
- Marks (including any penalties) awarded only if all test cases are passed
- Highlights minor formatting errors

For Codex

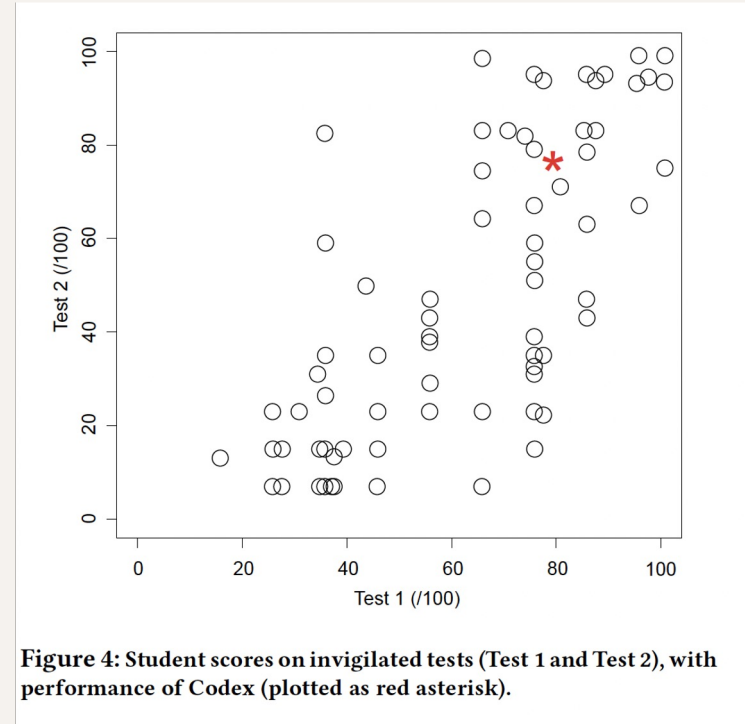
- Subjected to the same test cases
- Up to 10 attempts
- For trivial formatting error, manually amended the response and counted as an extra submission

Codex's Performance

- Of the 23 questions, 19 were solved within 10 responses.
- 10 questions were solved successfully on the first attempt (including correct response with trivial formatting error)
- 4 responses is correct but with trivial formatting error.

Comparison to Student Performance

- Codex scored 15.7/20 (78.5%) for Test 1 and 19.5/25 (78.0%) for Test 2.
- Codex's score is in position 17 when ranked alongside the 71 students' scores.



Challenges Encountered by Codex

Problem Constraints

- Q10 requires using while loop and prohibit use of `split()` method.
 - All answers fail to produce correct answer
- Q11 tasked with printing 4 integer in sorted order using only `min()` and `max()`.
 - Only 2 of the solutions produced correct answer but violate the constraints

Specific Formatting Requirements

- Q7 asked for an isosceles triangle print out based on integer input. Codex struggled with the precise formatting, particularly with leading spaces.
- Q12 required generating an ASCII histogram from a dictionary of bar heights. Codex's attempts fails in accurately representing the required format.

The Rainfall Problem

Reference	Problem wording
Soloway [39]	Write a program that will read in integers and output their average. Stop reading when the value 99999 is input.
Ebrahimi [9]	Write a program that will read the amount of rainfall for each day. A negative value of rainfall should be rejected, since this is invalid and inadmissible. The program should print out the number of valid recorded days, the number of rainy days, the rainfall over the period, and the maximum amount of rain that fell on any one day. Use a sentinel value of 9999 to terminate the program.
Simon [37]	A program has a one-dimensional array of integers called iRainfall, which is used to record the rainfall each day. For example, if iRainfall[0] is 15 and iRainfall[1] is 0, there was 15mm of rain on the first day and no rain on the second day. Negative rainfall values are data entry errors, and should be ignored. A rainfall value of 9999 is used to indicate that no more rainfall figures have been registered beyond that element of the array; the last actual rainfall value recorded is in the element immediately before the 9999. The number of days represented in the array is open-ended: it might be just a few days, or even none; it might be a month; it might be several years. The number of days is determined solely by the location in the array of the 9999 entry. Write a function method to find and return the average rainfall over all the days represented in the array. A day with negative rainfall is still counted as a day, but with a rainfall of zero.
Fisler [10]	Design a program called rainfall that consumes a list of numbers representing daily rainfall amounts as entered by a user. The list may contain the number -999 indicating the end of the data of interest. Produce the average of the non-negative values in the list up to the first -999 (if it shows up). There may be negative numbers other than -999 in the list.
Guzdial et al. [16], cited in [15]	Write a function rainfall that will input a list of numbers, some positive and some negative, e.g., [12, 0, 41, -3, 5, -1, 999, 17]. These are amounts of rainfall. Negative numbers are clearly a mistake. Print the average of the positive numbers in the list. (Hint: The average is the total of the positive numbers divided by the number of just the positive numbers.)
Lakanen et al. [18]	Implement the 'Average' function, which takes the amounts of rainfall as an array and returns the average of the array. Notice that if the value of an element is less than or equal to 0 ('lowerLimit'), it is discarded, and if it is greater than or equal to 999 ('sentinel'), stop iterating (the sentinel value is not counted in the average) and return the average of counted values.
<i>apples</i>	Create a method called harvest that takes one parameter that is a list of integers representing daily tonnes of fruit picked at a given orchard. It returns a floating point number rounded to 1 decimal place representing an average of the non-negative amounts up to either the first sentinel or the end of the list, whichever comes first. The sentinel is -999. If it is not possible to compute an average, then return -1.0. It is not possible to compute an average if there is no valid list (i.e. the parameter is None), or there are no non-negative values before the sentinel. There may be values after the sentinel but they are to be ignored when determining the average.

Evaluation Methodology

- No example test cases provided
- Each version of problem wording evacuate 50 times
- Total of 350 responses, each against 10 test cases
- Each response is marked out of 1 with each test cases contribute 0.1
- Record the number of passed tests and high-level matrix

Name	Stdin	List Argument
Blank	S\	[S]
One 0	0\S\	[0, S]
One +ve	5\S\	[5, S]
One -ve	-5\S\	[-5, S]
Multiple +ve	3\5\7\S\	[3, 5, 7, S]
Multiple 0's	0\0\0\S\	[0, 0, 0, S]
Multiple -ve	-3\ -5\ -7\S\	[-3, -5, -7, S]
Mixed +ve & 0	4\0\S\	[4, 0, S]
Mixed +ve & -ve	3\ -2\5\S\	[3, -2, 5, S]
Mixed All	3\0\ -2\5\S\	[3, 0, -2, 5, S]

Table 3: Rainfall test cases (S→Sentinel, \→newline).

Codex's Performance on Rainfall

Variants

- Outperformed Simon variant students (29% average partial score) with C#.
- Similar performance to Guzdial et al. Python students (46%).
- Lower performance compared to Lakanen et al. variant students (69%).

However, the test cases used in above published variant is different. Not a fair comparison.

Variant	Mean	Median	Max	Stddev
Soloway [39]	0.63	0.90	1.00	0.40
Simon [37]	0.48	0.50	1.00	0.28
Fisler [10]	0.61	0.70	1.00	0.26
Ebrahimi [9]	0.19	0.05	1.00	0.26
Guздial et al. [16]	0.47	0.30	1.00	0.22
Lakanen et al. [18]	0.44	0.70	0.90	0.32
<i>apples</i>	0.54	0.60	1.00	0.34

Table 4: Rainfall results marked out of 1 (max score).

Direct Comparison of the Apple Variant

Students' Performance

- Same test cases
- 45 CSI level students, received an average partial mark of 84%
- Allow resubmission

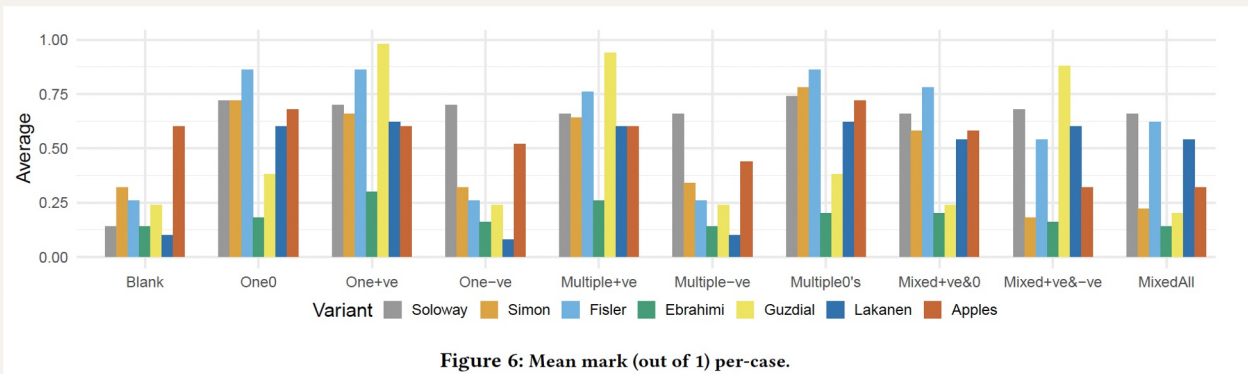
Codex's Performance

- Lower score (54%)

Variant	Mean	Median	Max	Stddev
Soloway [39]	0.63	0.90	1.00	0.40
Simon [37]	0.48	0.50	1.00	0.28
Fisler [10]	0.61	0.70	1.00	0.26
Ebrahimi [9]	0.19	0.05	1.00	0.26
Guzdial et al. [16]	0.47	0.30	1.00	0.22
Lakanen et al. [18]	0.44	0.70	0.90	0.32
<i>apples</i>	0.54	0.60	1.00	0.34

Table 4: Rainfall results marked out of 1 (max score).

Challenge Identified



Name	Stdin	List Argument
Blank	S\	[S]
One 0	0\S\	[0, S]
One +ve	5\S\	[5, S]
One -ve	-5\S\	[-5, S]
Multiple +ve	3\5\7\S\	[3, 5, 7, S]
Multiple 0's	0\0\0\S\	[0, 0, 0, S]
Multiple -ve	-3\ -5\ -7\S\	[-3, -5, -7, S]
Mixed +ve & 0	4\0\S\	[4, 0, S]
Mixed +ve & -ve	3\ -2\5\S\	[3, -2, 5, S]
Mixed All	3\0\ -2\5\S\	[3, 0, -2, 5, S]

Table 3: Rainfall test cases (S→Sentinel, \→newline).

Solution Diversity

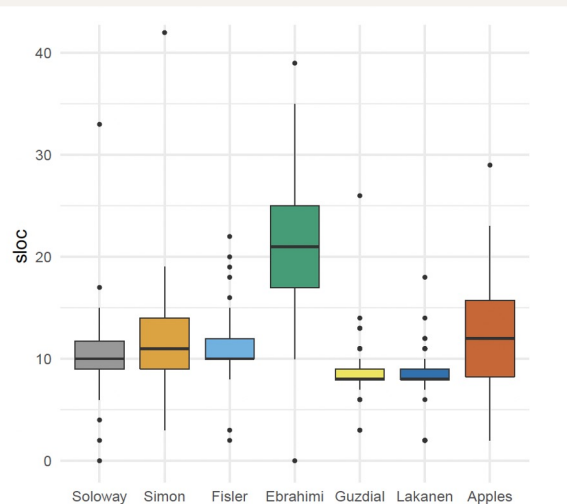


Figure 7: Source lines of code (sloc) per variant.

Variant	One while	One for	Sum / Len	Two Pass	Other
Soloway [39]	41	1	0	5	3
Simon [37]	5	36	1	8	0
Fisler [10]	3	42	1	3	1
Ebrahimi [9]	36	3	0	10	1
Guzdial et al. [16]	0	35	1	13	1
Lakanen et al. [18]	2	37	2	5	4
<i>apples</i>	4	23	3	16	4

Table 5: Count of general method used by response.

examined the number of source lines of code (sloc) excluding blank and comment lines

Limitations

- Training Data Concerns
 - Codex's training may include solutions to variants of the Rainfall Problem, potentially influencing its performance.
 - Solutions from test questions could have been posted online, becoming part of Codex's training dataset.
- Unique Variant Mitigation: Utilization of a novel "apples" variant aimed to minimize prior knowledge effects.
- Creative Response Setting: High temperature value (0.9) used to encourage Codex to generate more varied, less predictable solutions.

Conclusion & Discussions

- Codex demonstrates capability far beyond that expected by the authors
- Opportunities
 - Codex can provide diverse solutions, offering rich learning and discussion opportunities.
 - Automatically generated answers may serve as learning aids, particularly for practice exercises without provided solutions.
- Challenges
 - Potential for dependency, with students possibly adopting incorrect solutions or poor coding practices.
 - Difficulty distinguishing AI-generated work from student efforts, complicating plagiarism detection and assessment fairness.
- Future Directions
 - Can automated plagiarism detection tools identify code generated by Codex?
 - Can tools like Codex be utilised to detect plagiarism?
 - How can Codex be used to improve student learning?
 - How should we adapt course content and assessment approaches as the use of tools such as Codex becomes more prevalent?

Conclusion

- Codex demonstrates capability far beyond that expected by the creators
- Having Codex in the hands of students should warrant concern similar to having a power tool in the hands of an amateur.
- The tool's current limitations under tight constraints are expected to diminish with rapid technological advancements.
- Codex offers opportunities to innovate curricula, including creating unique exam content and exposing students to a variety of solutions.
- The AI revolution in education demands adaptation, with Codex at the forefront of transforming how programming is taught and assessed.
- Embracing this change is not optional; proactive adaptation and discussion on ethical AI use in education are crucial for future success.

Thanks