מסייע למאמן: סוכנים כסימולטורים להערכה אותנטית בהנדסת תוכנה

אסף ב. שפנייר, **ליאור קץ, יעל בוכמן, מיכאל לוגסי, רפאל זנזורי, שרה עייש ויוכבד רוטשטיין רוט –**

עזריאלי, מכללה אקדמית להנדסה ירושלים

Production Trap Simulator

· הגשה 1 מתוך 3 · יעל בוכמן + ליאור קץ ·

סימולטור אג'נטי שמכניס את הסטודנט למחזור-חיים אמיתי של תקלת production — דרישות מעורפלות, חיכוך מקצועי, ותקלות מערכת קריטיות שכלי AI גנרי מתקשה לפתור בצעד אחד. במקום תרגיל קוד רגיל, המערכת פועלת כ«מחולל בעיות».

- מעבר ל-syntax — יציבות מערכת ו-production
- תגובה לאירועים (incident response)
- תקשורת מקצועית תחת לחץ

Production Trap — אדריכלות רב-סוכנית (LangGraph)

Senior Team Lead
מאשר קוד ש«עובד» אך אינו production-ready ← מפעיל deployment.

Production Monitor
מזהה בשלים קריטיים (DB locks, API rate limits) ומדווח באירוע.

Software Architect
post-mortem; דורש דפוסים (non-blocking DDL, caching) לפני «נפתר».

עמידות ל-AI: מוטציית תרחישים דינמית — שמות טבלאות/משתנים משתנים בבל הרצה, כך ש-60+ סטודנטים לא יכולים לשתף פתרונות. · *o4Python · LangGraph · TypedDict + MemorySaver · GPT-*

Legacy Code Challenge

· הגשה 2 מתוך 3 · רפאל זנזורי · מיכאל לוגסי

מערכת מבוססת-AI שהופכת כל ריפו GitHub ציבורי לאתגר דיבוג חי: משכפלת ריפו, מזריקה באגים ריאליסטיים לשרשראות קריאה עמוקות, ומגישה לסטודנט ממשק Gradio עם עורך קוד, מריץ בדיקות, עוזר רמזים חכם, ומעריך הגשות מבוסס-LLM.

- דיבוג בקוד לגאסי אמיתי
- אתגר מותאם-קושי (nesting-level, num-bugs)
- בדיקות סמויות + גלויות

Cloud Migration Simulation

· הגשה 3 מתוך 3 · שרה עייאש ויובבד רוטשטין·

סימולציה שיחתית לתרגול קבלת החלטות בהגירת ענן: המשתמש מתכנן מעבר של קוד AWS לספק ענן חלופי, בשיחה קבוצתית מול פרסונות (PM, DevOps, CTO) בממשק Streamlit — תחת אילוצי זמן, עלות, אבטחה, ביצועים ו-downtime.

- החלטות ארכיטקטורה תחת אילוצים
- מגוון נקודות-מבט מקצועיות
- משוב וציון מיידיים (0–10)

Cloud Migration — פרסונות, תרחישים והערכה

פרסונות

PM (לחץ זמן) · DevOps (אבטחה/תשתית) · CTO (עלות/אסטרטגיה) — פרסונה שונה בכל סבב.

תרחישים

קומבינציות AWS אקראיות (SNS+3S, SQS+adbmaL, MAI+BDomanyD) + הקשר עסקי + אילוצים.

הערכה

ציון 0–10, חוזקות, פערים והמלצות; סבב review מסכם; אנימציית סיום לפי הציון.

המערכת מוודאת שה-API והמודל עובדים לפני הסימולציה, ומבטיחה גיוון פרסונות. · Python · OpenAI / Anthropic (gpt-o-mini) · Streamlit4

VeriExam

לתת לסטודנטים להשתמש ב-AI — בלי לתת ל-AI ללמוד במקומם.

שני מצבי הכשל של אינטגרציית LLM נאיבית

מכונת התשובות האוטומטית
הפרומפט הראשון ← פתרון מלא ← העתק-הדבק ← המבחן עובר ← שום דבר לא נלמד.
הורס את התרגיל.

הססן הבלתי-נראה
למרצה אין שום מבט על איך נעשה שימוש במודל — נתקע? הוטעה? תשובה בשלוף?
הורס את ההערכה.

תנו לכל סטודנט חלון צ׳אט גולמי — ותקבלו אחד משני אלה. שניהם מבוי סתום פדגוגי.

המערכת במבט-על

- לוח הבקרה של המרצה — React + Vite + TS (port 5173)
- Backend — FastAPI + SQLite WAL (port 8010)
- צ’אט הסטודנט — Gemini (streaming)
- קליינט סטודנט A — Chainlit web app (port 8011)
- קליינט סטודנט B — CLion plugin, Kotlin (in-IDE)
- שכבת הסוכנים — CrewAI on Gemini

Backend אחד, שני קליינטים חליפיים — דפדפן או תוסף IDE, אותה חוויה.

main.py | mcalc.c | pycalc.py

No Python interpreter configured for ... Create a virtual environment Custom Environment

```
skeleton below reads that line and hands you the quoted text as `arg`

The OS Lab tutor (right-hand tool window) can help — but it will not
hand you a finished solution.
"""

import ...


def solve(arg: str) -> str:
    """Compute the answer for `arg` and return it as a string
    (or the exact required error string)."""
    # TODO: your code here
    raise NotImplementedError


if __name__ == "__main__":
    line = sys.stdin.readline()
    m = re.search(r'"(.*)"', line)          # the text between the quotes
    arg = m.group(1) if m else line.strip()
    print(solve(arg))
```

OS Lab

⚠ Exam mode — You switched away from PyCharm.
Reported to your instructor. You have left the exam window 1 time(s). Got it

Yossi Cohen · variant V-P1 · mode mentor · client ide Send to teacher

pycalc — Built-in Arithmetic Calculator (Python) (variant V-P1)

In this exercise you will build a built-in command called `pycalc` in **Python**. Like a mini-shell built-in, it reads a single command line from **standard input** of the form:

```
pycalc "3 + 4 * 2"
```

It parses the arithmetic expression, computes the result, and prints the **integer result on its own line** (the last line of output is what is graded):

```
pycalc "3 + 4 * 2"
11
```

Read the whole line from `stdin`, take the text between the quotes as the expression, and print the answer. Your `pycalc` must respect operator precedence (`*`, `/`, `%` before `+` and `-`) and the specific parameters of your assigned variant (Allowed operators: +, -, *, /, Parentheses allowed: no, Maximum result: 100). Using an operator that is not allowed, or parentheses when they are not allowed, is an invalid expression.

Send

VeriExam — לוח בקרה למורה

התנתקות | English 🌐

מעקב חי | תרגילים

תרגילים

+ תרגיל חדש

ex3-mcalc מובנה (נערך)
תרגיל בדיקה
1 וריאנטים · מנטור
הפוך לפעיל | עריכה | שכפול

ex4-mstat מובנה
mstat — סטטיסטיקה על רשימת מספרים
6 וריאנטים · מנטור
הפוך לפעיל | עריכה | שכפול

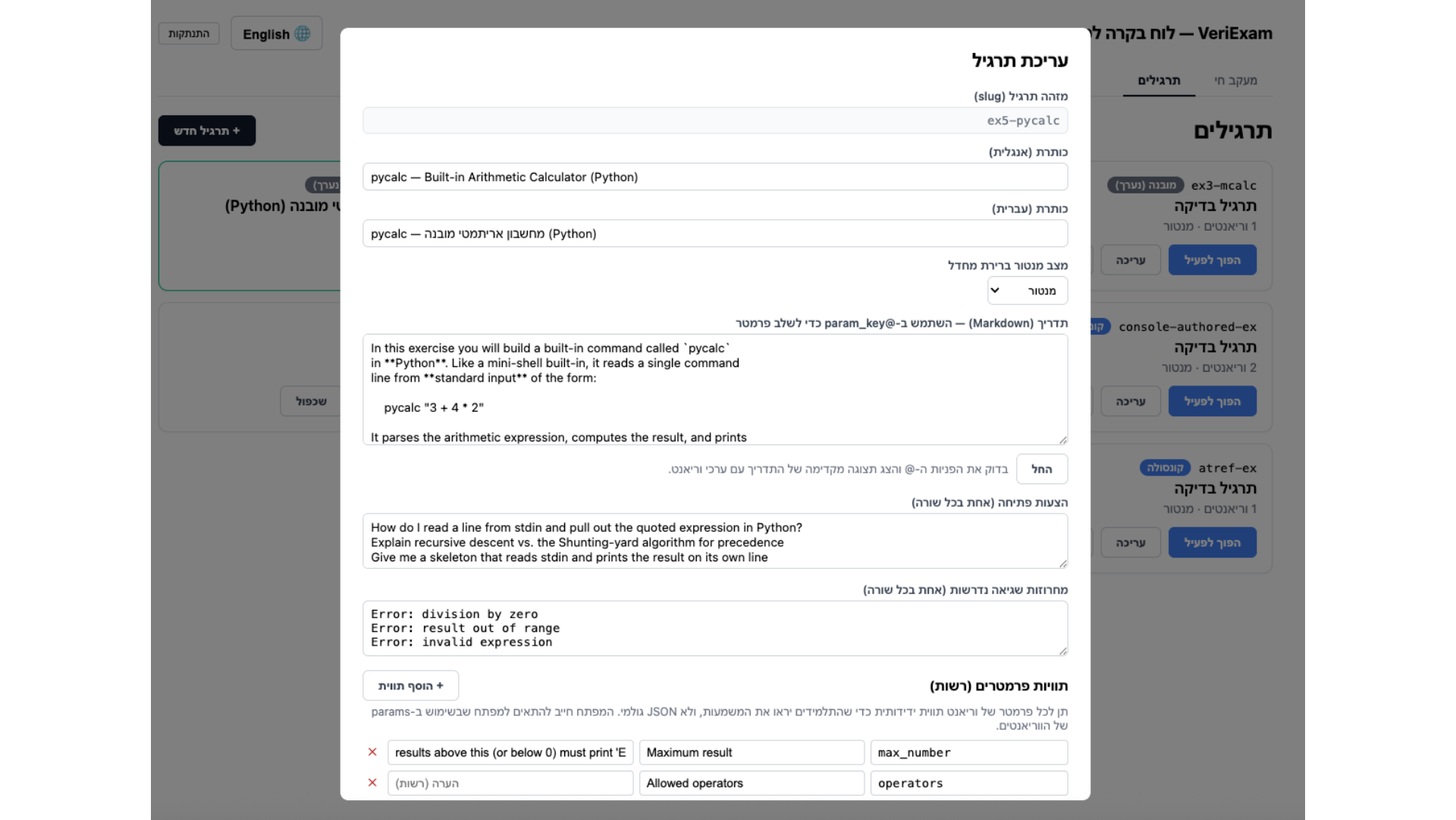
ex5-pycalc פעיל מובנה (נערך)
pycalc — מחשבון אריתמטי מובנה (Python)
6 וריאנטים · מנטור
עריכה | שכפול

console-authored-ex קונסולה
תרגיל בדיקה
2 וריאנטים · מנטור
הפוך לפעיל | עריכה | שכפול

ex3-mcalc-copy קונסולה
mcalc — מחשבון אריתמטי מובנה
8 וריאנטים · מנטור
הפוך לפעיל | עריכה | שכפול

labeled-ex קונסולה
תרגיל בדיקה
2 וריאנטים · מנטור
הפוך לפעיל | עריכה | שכפול

atref-ex קונסולה
תרגיל בדיקה
1 וריאנטים · מנטור
הפוך לפעיל | עריכה | שכפול


התנתקות
English
VeriExam — לוח בקרה ל
מעקב חי
תרגילים
+ תרגיל חדש
תרגילים
ex3-mcalc
מובנה (נערך)
תרגיל בדיקה
1 וריאנטים · מנטור
הפוך לפעיל
עריכה
console-authored-ex
תרגיל בדיקה
2 וריאנטים · מנטור
הפוך לפעיל
עריכה
atref-ex
קונסולה
תרגיל בדיקה
1 וריאנטים · מנטור
הפוך לפעיל
עריכה
שכפול
עריכת תרגיל
מזהה תרגיל (slug)
ex5-pycalc
כותרת (אנגלית)
pycalc — Built-in Arithmetic Calculator (Python)
כותרת (עברית)
pycalc — מחשבון אריתמטי מובנה (Python)
מצב מנטור ברירת מחדל
מנטור
תדריך (Markdown) — השתמש ב-@param_key כדי לשלב פרמטר
In this exercise you will build a built-in command called `pycalc`
in **Python**. Like a mini-shell built-in, it reads a single command
line from **standard input** of the form:
pycalc "3 + 4 * 2"
It parses the arithmetic expression, computes the result, and prints
החל
בדוק את הפניות ה-@ והצג תצוגה מקדימה של התדריך עם ערכי וריאנט.
הצעות פתיחה (אחת בכל שורה)
How do I read a line from stdin and pull out the quoted expression in Python?
Explain recursive descent vs. the Shunting-yard algorithm for precedence
Give me a skeleton that reads stdin and prints the result on its own line
מחרוזות שגיאה נדרשות (אחת בכל שורה)
Error: division by zero
Error: result out of range
Error: invalid expression
תוויות פרמטרים (רשות)
+ הוסף תווית
תן לכל פרמטר של וריאנט תווית ידידותית כדי שהתלמידים יראו את המשמעות, ולא JSON גולמי. המפתח חייב להתאים למפתח שבשימוש ב-params של הווריאנטים.
results above this (or below 0) must print 'E
Maximum result
max_number
הערה (רשות)
Allowed operators
operators

תרגיל פעיל: ex5-pycalc

pycalc — מחשבון אריתמטי מובנה (Python) · 6 וריאנטים · 9 תלמידים מחוברים

מצב בדיקת מורה כבוי — התלמידים מקבלים תשובות ישירות מה-LLM.

כבוי

sim★ Rote but correct · ex5-pycalc

ex5-pycalc V-P1 מנטור 3

85

Maximum result: **100** · Allowed operators: **+, no :-, *, /** · Parentheses allowed

פעילות אחרונה: 04:41:21 2026-07-15

מנטור · צפה בצ'אט · שאל אותם · הגשות (1)

sim★ Compile failed but engaged · ex5-pycalc

ex5-pycalc V-P1 מנטור 3

64

Maximum result: **100** · Allowed operators: **+, no :-, *, /** · Parentheses allowed

פעילות אחרונה: 04:38:21 2026-07-15

מנטור · צפה בצ'אט · שאל אותם · הגשות (1)

sim★ Diligent average · ex5-pycalc

ex5-pycalc V-P1 מנטור 3

96

Maximum result: **100** · Allowed operators: **+, no :-, *, /** · Parentheses allowed

פעילות אחרונה: 04:35:21 2026-07-15

מנטור · צפה בצ'אט · שאל אותם · הגשות (1)

sim★ Cheater (pasted external) · ex5-pycalc

ex5-pycalc V-P1 מנטור 3

34

Maximum result: **100** · Allowed operators: **+, no :-, *, /** · Parentheses allowed

פעילות אחרונה: 04:32:21 2026-07-15

מנטור · צפה בצ'אט · שאל אותם · הגשות (1)

sim★ Gave up (empty) · ex5-pycalc

ex5-pycalc V-P1 מנטור 3

4

Maximum result: **100** · Allowed operators: **+, no :-, *, /** · Parentheses allowed

פעילות אחרונה: 04:29:21 2026-07-15

מנטור · צפה בצ'אט · שאל אותם · הגשות

sim★ Independent (minimal LLM) · ex5-pycalc

ex5-pycalc V-P1 מנטור 3

62

Maximum result: **100** · Allowed operators: **+, no :-, *, /** · Parentheses allowed

פעילות אחרונה: 04:26:21 2026-07-15

מנטור · צפה בצ'אט · שאל אותם · הגשות (1)

sim★ Hint spammer · ex5-pycalc

ex5-pycalc V-P1 מנטור 3

48

Maximum result: **100** · Allowed operators: **+, no :-, *, /** · Parentheses allowed

פעילות אחרונה: 04:23:21 2026-07-15

מנטור · צפה בצ'אט · שאל אותם · הגשות (1)

sim★ Engaged but stuck · ex5-pycalc

ex5-pycalc V-P1 מנטור 3

96

Maximum result: **100** · Allowed operators: **+, no :-, *, /** · Parentheses allowed

פעילות אחרונה: 04:20:21 2026-07-15

מנטור · צפה בצ'אט · שאל אותם · הגשות (1)

📘 תרגיל פע

pycalc — מחשבון אריתמ

מצב בדיקת מורה

כבוי

sim★ Cheate
external) · ex

Maximum result: 100 · Allo
no :-, *, / · Pa
04:32:21 2026-07-

צפה בצ'אט
הגשות (1)

sim★ Engaged bu
ex

Maximum result: 100 · Allo
no :-, *, / · Pa
04:20:21 2026-07-

צפה בצ'אט
הגשות (1)

te but correct · SR
ex5-pycalc
V-P1 | ex5-pycalc
85
100 · Allowed operators: +,
, *, / · Parentheses allowed
פעילות אחרונה: 2026-07-15
מנטור
שאל אותם

ve up (empty) · SG
ex5-pycalc
V-P1 | ex5-pycalc
4
100 · Allowed operators: +,
, *, / · Parentheses allowed
2026-07-15
מנטור
שאל אותם

and-play copier SP
· ex5-pycalc
מנטור | V-P1 | ex5-pycalc

sim★ Diligent average · ex5-pycalc · V-P1 · mentor

Maximum result: 100 | Allowed operators: +, -, *, / | Parentheses allowed: no

גלילה אוטומטית | סגור

user · 2026-07-14 20:38:58 · mentor
אפשר להשתמש ב `eval`?

assistant · 2026-07-14 20:39:04 · mentor
שאלה טובה. הפונקציה `eval` היא אכן כלי חזק מאוד בפייתון שיכול לפתור את התרגיל הזה בקלות. עם זאת, שים לב שהקוד שכתבת הוא ב-C, שפה שבה אין פונקציית `eval` מובנית לביטויים אריתמטיים. אני ממליץ לך לעיין שוב במפרט התרגיל ולבדוק באיזו שפת תכנות נדרש להשתמש.

user · 2026-07-14 20:41:04 · mentor
איך אני קורא רק מה שבתוך המרכאות?

assistant · 2026-07-14 20:41:10 · mentor
היי, זו שאלה מצוינת. למעשה, הקוד שכתבת בתוך פונקציית ה-`main` שלך כבר עושה בדיוק את זה!

שים לב איך אתה משתמש ב-`strncmp` כדי לוודא שהשורה מתחילה ב-`pycalc "`. לאחר מכן, אתה מקדם מצביע בשם `expr_start` כך שיצביע בדיוק אחרי המרכאה הפותחת. לבסוף, אתה משתמש ב-`strrchr` כדי למצוא את המרכאה הסוגרת ומחליף אותה בתו `\0`, מה שגורם למחרוזת להסתיים שם. זו טכניקה נכונה ויעילה ב-C לבידוד תת-מחרוזת.

user · 2026-07-14 20:43:10 · mentor
ואיך אני בודק את ה `overflow`?

assistant · 2026-07-14 20:43:16 · mentor
היי, שאלה טובה. כרגע הקוד שלך בודק אם המספרים המקוריים ב-`input` חורגים מהטווח המותר, וזה התחלה מצוינת.

כדי לטפל ב-`overflow`, עליך להוסיף בדיקה דומה גם *אחרי* כל פעולה חשבונית שאתה מבצע. לדוגמה, אחרי שחישבת את התוצאה של `a * b`, ולפני שאתה שומר אותה, בדוק אם הערך המוחלט שלה (`labs()`) גדול מ-`MAX_NUMBER`. אם כן, הדפס את `ERR_RANGE` ועצור את המשך החישוב.

בדיקת מרצה · suggested · 2026-07-14 20:44:46 · answered

העזר ציין שהקוד שלך משתמש ב-`strrchr` כדי למצוא את המרכאה הסוגרת ולהחליף אותה בתו `\0`. מה היה קורה להתנהגות התוכנית אם היית משתמש בפונקציה `strchr` במקום זאת, ועבור איזה קלט ספציפי היינו רואים את ההבדל?

תשובה · ציון 2/2

אם הייתי משתמש ב-`strchr`, הוא היה מוצא את המרכאה הסוגרת הראשונה שהוא פוגש. `strrchr` מוצא את האחרונה. היינו רואים את ההבדל בקלט כמו `"quote" and more text "pycalc" 5 + 10`. במקרה כזה, `strchr` היה מוצא את המרכאה שאחרי ה-5 והיה מחשב נכון את `5 + 10`. אבל הקוד שלי עם `strrchr` ימצא את המרכאה האחרונה בשורה וינסה לחשב את כל הדבר `"quote" and more text "pycalc" 5 + 10`, מה שכנראה היה נכשל כי יש שם אותיות.

6 הודעות צ'אט · 3 בדיקות · מתרענן כל 2 שניות

📘 תרגיל פעיל: ex5-pycalc

pycalc — מחשבון אריתמטי מובנה (Python) · 6 וריאנטים · 8 תלמידים מחוברים

👩‍🏫 מצב בדיקת מורה שאלות התלמידים ממתינות עד שתערוך ותשלח תשובה.

פעיל ☑

Yossi Cohen · 2026-07-11 21:21:19

Explain how to parse an arithmetic expression

תשובות מוצעות בשפה: עב | EN

רמז

מכיוון שה-`mcalc` שלך חייב לכבד קדימות אופרטורים, אילו אופרטורים (`*`, `/` או `+`, `-`) הקוד שלך צריך לחפש ולחשב ראשונים כשהוא מנתח ביטוי כמו "3 + 4 * 2"?

שלח · רמז

רגיל

מכיוון שה-`mcalc` שלך חייב לכבד קדימות אופרטורים, אילו אופרטורים (`*`, `/` או `+`, `-`) הקוד שלך צריך לחפש ולחשב ראשונים כשהוא מנתח ביטוי כמו "3 + 4 * 2"?

אסטרטגיה נפוצה לכך היא גישה של שני מעברים. ראשית, סרוק את המחרוזת כדי למצוא ולחשב את כל פעולות הכפל והחילוק. לאחר מכן, במעבר שני על תוצאות הביניים, חשב את

שלח · רגיל

רגיל + הרחבה

מכיוון שה-`mcalc` שלך חייב לכבד קדימות אופרטורים, אילו אופרטורים (`*`, `/` או `+`, `-`) הקוד שלך צריך לחפש ולחשב ראשונים כשהוא מנתח ביטוי כמו "3 + 4 * 2"?

אסטרטגיה נפוצה לכך היא גישה של שני מעברים. ראשית, סרוק את המחרוזת כדי למצוא ולחשב את כל פעולות הכפל והחילוק. לאחר מכן, במעבר שני על תוצאות הביניים, חשב את כל

שלח · רגיל + הרחבה

שאלה ממוקדת עבור · ex5-pycalc · sim★ Diligent average · V-P1

תשובות מוצעות בשפה: עב EN

[Failure-mode]

העוזר צריך שהקוד שלך משתמש ב-`strrchr` כדי למצוא את המרכאה הסוגרת. מה יקרה אם משתמש יריץ את התוכנית עם הקלט `pycalc "5 + 3` (כלומר, ללא מרכאה סוגרת)? באיזו נקודה בקוד שהעוזר תיאר תתרחש הבעיה, ומה יהיה סוג הכשל (למשל, קריסה, פלט שגוי)?

בודק הבנה של מקרי קצה בטיפול במחרוזות ב-C, ספציפית מה `strrchr` מחזיר כשלא נמצא תו ומה קורה כשמנסים לעבוד עם מצביע NULL.

[Why]

העוזר הציע לבדוק חריגה מהטווח *אחרי* כל פעולה. נניח שיש לך את הביטוי `60 + 60`. שני המספרים, 60 ו-60, נמצאים בטווח המותר (`max_number: 100`). מדוע לא מספיק לבדוק רק את המספרים שמופיעים בקלט המקורי, וצריך לבדוק גם את תוצאות הביניים?

מוודא שהסטודנט מבין שפעולות אריתמטיות יכולות ליצור תוצאות שחורגות מהטווח, גם אם האופרנדים עצמם תקינים.

[Trace]

בהתבסס על תיאור העוזר, הקוד שלך מקדם מצביע `expr_start` לתחילת הביטוי. עבור הקלט `pycalc "-20 + 5"`, כיצד המנתח שלך אמור להבדיל בין התו '-' שבתחילת הביטוי (כסימן של מספר שלילי) לבין אותו תו כשהוא מופיע כאופרטור חיסור (למשל, בביטוי `30 - 10`)?

בוחן את ההבנה של אתגר נפוץ בניתוח ביטויים – ההבחנה בין אופרטור אונארי (סימן) לבינארי (פעולה) – נושא ש-`eval` פותר באופן שקוף.

הצע עוד

כתוב שאלה משלך:

שלח 0 שאלות | ביטול

הגשות · V-P1 · ex5-pycalc · sim★ Hint spammer

סגור

1 הגשות · מתרענן כל 3 שניות

ציון מומלץ

מסומן — העתק-הדבק?

ט חשב מחדש

48 / 100 נכונות **100** שימוש נבון ב-LLM **25**

נימוק: הסטודנט הגיש פתרון מלא בשפת C לתרגיל שדורש פתרון בפייתון, תוך התעלמות מוחלטת מהנחיית הבודק האוטומטי. האינטראקציה עם ה-AI מאופיינת בשאלות כלליות, הסתמכות על רמזים ישירים, וזמני תגובה מהירים באופן מחשיד המצביעים על חוסר למידה אמיתי. למרות שהסטודנט הפגין הבנה של האלגוריתם שכתב ב-C, הוא נכשל לחלוטין בהבנת הרעיון המרכזי של השימוש ב-`eval()` בפייתון, כפי שהודגם בתשובתו השגויה לשאלת ההבנה השלישית.

- שאלה ראשונית כללית מאוד ("איך אני אמור לעשות את זה בכלל?") המעידה על חוסר ניסיון עצמאי.
- התעלמות עקבית ומוחלטת מההנחיה המרכזית של המדריך לעבור לפייתון ולהשתמש ב-`eval()`.
- הגשת פתרון בשפת C לתרגיל `pycalc`, שמטרתו ללמד על יכולות ספציפיות בפייתון.
- זמן קצר מאוד של 3 שניות בין קבלת הרמז האחרון להגשת הפתרון, מה שמצביע על כך שהקוד כבר היה מוכן והרמז לא עזר בתהליך.
- כישלון מוחלט במענה על שאלת הבנה מס' 3, שחשף חוסר הבנה בסיסי של `eval()` (הסטודנט ענה בהקשר של קוד ה-C שלו).
- הפגנת הבנה טובה של אלגוריתם ה-parser שהסטודנט מימש בעצמו בשפת C (תשובות נכונות לשאלות 1 ו-2).

עקיפת מורה: 0–100 שמור

pycalc.c · c

2026-07-14 20:31:59

✓ 3/3 ✓ ✓ ✓

קוד מקור

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>

} ()int main
;char line[2048]

} while (fgets(line, sizeof(line), stdin) != NULL)
    ?how do I do this, just check the start //
    } if (strncmp(line, "pycalc \"", 8) != 0)
                ;continue
                    {

;char *expr = line + 8
```

פלט קונסול

```
$ pycalc "2 + 3" -> 5
$ pycalc "50 * 3" -> Error: result out of range
$ pycalc "10 / 0" -> Error: division by zero
```

מצב 1: פרסונה (Persona)

mentor — סוקרטי
מחזיר שאלות, מסרב לקוד מלא, מפנה למושגים. ברירת מחדל בשלב מוקדם.

coding_partner — שיתופי
כותב מקטעים קטנים, עדיין ללא פתרון גמור. ברירת מחדל לאחר שהודגמה הבנה.

אותה שאלה, שני מצבים: mentor עונה בשאלה, coding_partner במקטע קוד. מתג system-prompt פר-סטודנט.

מצב 2: בדיקות הבנה (Comprehension probes)

1. **פתיחת Ask Them על כרטיס סטודנט**
   המרצה מקליד שאלה — או לוחץ Suggest.

2. **Suggest = הצעה אוטומטית**
   ה-LLM קורא את התמליל האחרון ומציע 2–3 בדיקות ממוקדות לפערים.

3. **הבדיקה קוטעת את הצ'אט**
   נבדקת אוטומטית 0–2 ונשמרת. המרצה יכול לעקוף את הציון.

בודק אם הסטודנט מבין את מה שה-AI עזר לו לייצר — במילים שלו, ברגע אמת. עמוד השדרה של ההערכה.

היכן זה עומד ומה הלאה

סטטוס:

פלטפורמת מחקר עובדת, בשימוש ל-Intro-to-OS ב-JCE; תרגיל דמו clacm-3ex (zero-8 ,(variants config.

מפת דרכים:

- פיגום אדפטיבי — בחירת answer-level מהיסטוריית ה-probe
- סשנים רב-תרגיליים
- זיכרון פר-סטודנט חוצה-סשנים
- אריזת Docker one-stack למחלקות אחרות

VeriExam

- מרצה בתוך הלולאה.
- וריאנטים שהופכים שיתוף לחסר-טעם.
- Probes שמעריכות הבנה — לא פלט.

לתת לסטודנטים להשתמש ב-AI — בלי לתת ל-AI ללמוד במקומם.