# מעבדה בבינה מלאכותית 2026
# 203.3630 .ב.1

## סמסטר ב' – שנה"ל תשפ"ו

## דגשים למעבדה

1. **אתיקה והגינות אקדמית** – הקוד חייב להיות מקורי ונכתב על־ידכם. קוד מועתק, או שימוש לא מוצהר בקוד חיצוני, ייפסלו והציון למגישים יהיה אפס, בהתאם לנהלי הקורס והפקולטה.
2. **שימוש במחוללי קוד** - שימוש בכלי AI ומחוללי קוד כגון ChatGPT, Claude, GitHub Copilot ודומיהם מותר רק ככלי עזר מוגבל, למשל לצורך הבהרת תחביר, דיבוג נקודתי, או שיפור ניסוח תיעוד.
3. **דרישת יכולת הסבר** - אין להגיש קוד שנוצר על ידי כלים אלה ללא הבנה מלאה של הסטודנטים וללא אחריות מלאה עליו.
כל שימוש מהותי בכלים כאלה חייב להיות מוצהר במפורש במסמך ההגשה, כולל פירוט קצר של אופי השימוש: המגישים חייבים להיות מסוגלים להסביר כל שורת קוד שהגישו. קוד שלא יוסבר כראוי ייחשב כאילו לא נכתב על ידי המגישים. כנ"ל לגבי התוצרים של הקוד. התוצרים והקוד חייבים להיות עקביים ומוסברים.
4. **עמידה בזמנים** – יש להגיש עד המועד שנקבע. איחור ללא אישור מראש ייחשב כאי־הגשה. האחריות להגשה תקינה ובזמן היא על הסטודנטים.
5. **עבודה בזוגות** – הזוגות ייקבעו בתחילת הסמסטר ויישארו קבועים עד סופו.
6. **פירוק זוג** – במקרה של פירוק זוג, ההגשות מאותו שלב ואילך יהיו אישיות, אלא אם יוחלט אחרת על ידי צוות הקורס.
7. **גודל קבוצה** – לא תתאפשר הגשה של יותר משני סטודנטים בשום תנאי.
8. **שפות תכנות מותרות** – כל שפה מקובלת. מומלץ להגיש באחת מהשפות הבאות בלבד: C, C++, C#, Java, **Python**, Julia.
9. **איכות קוד** – הקוד צריך להיות מובנה, מודולרי, קריא ומתועד היטב. כאשר מתאים, מומלץ שימוש בתכנות מונחה עצמים.
10. **ניידות והרצה** – הקוד צריך להיות ניתן לקימפול/הרצה על מחשב הבודק, ללא תלות בנתיבים מקומיים או בהגדרות ייחודיות למחשב שלכם. כל תלות לא סטנדרטית חייבת להיות מפורטת בצורה מלאה.

## הנחיות ספציפיות

1. **אין לקודד נתיבים מקובעים** למחשב שלכם או לסביבת העבודה שלכם. נתיב לקובץ קלט חייב להינתן כארגומנט בשורת הפקודה. יש לציין בצורה ברורה במסמך ההגשה כיצד מריצים את התוכנית. לדוגמה:
```
java RushHour <path_to_input_file> [max_time_in_seconds]
python rush_hour.py <path_to_input_file> [max_time_in_seconds]
```
2. **זמן ריצה מקסימלי** –אם נדרש timeout, יש לאפשר לקבוע אותו כארגומנט, ולא לקודד אותו בתוך התוכנית .
3. **תיעוד הקוד והפתרון** –יש לתעד לפחות כל פונקציה מרכזית בקוד. בנוסף, בדו"ח יש להסביר את המימוש, מבנה המערכת, והארכיטקטורה של הפתרון .

4. **שפת הערות** –כל ההערות בקוד חייבות להיות באנגלית בלבד .

5. **שפות מתפרשות** –בעבודות בשפות כגון Python יש לציין במפורש את גרסת המפענח/האינטרפרטר שבה השתמשתם, וכן לפרט את כל החבילות שאינן חלק מהספרייה הסטנדרטית, כולל מספרי גרסה .

6. **שפות מתקמפלות** –בעבודות בשפות כגון C/C++/C#/Java יש לציין במפורש הוראות קימפול, דגלי קימפול מיוחדים, וספריות חיצוניות אם ישנן .

7. **קבצים בינאריים** –אין צורך להגיש קבצים בינאריים כגון pyc, class, או קובצי executable, אלא אם צוין אחרת במפורש .

8. – **Makefile / build script**השימוש ב־ Makefile או בסקריפט build מותר ואף מומלץ. אם נעשה בו שימוש, יש לכלול בו את כל חוקי הקימפול והקישור הנדרשים .

9. **טיפול בשגיאות צפויות** –במקרה של שגיאות קלט או שגיאות צפויות אחרות שאינן קריטיות, התוכנית צריכה לדווח בצורה ברורה מה הבעיה והיכן היא התרחשה, ולא לקרוס ללא הסבר. מטרת הדיווח היא לאפשר להבין אם מקור הבעיה הוא בקלט או במימוש .

10. **ללא ממשק אינטראקטיבי** –יש להימנע מתפריטים אינטראקטיביים ובקשות קלט מהמשתמש בזמן ריצה. כל הקלטים הדרושים צריכים להינתן דרך שורת הפקודה, כפי שמוגדר בסעיף 1 .

11. **הוראות הרצה** –יש לצרף קובץ README קצר וברור הכולל :

- הוראות קימפול/הרצה
- תלויות וספריות חיצוניות
- פורמט קלט נדרש
- דוגמה אחת לפחות להרצה תקינה

12. **אקראיות ושחזור** –אם הפתרון כולל רכיב אקראי, יש לאפשר קביעת seed או לציין כיצד ניתן לשחזר תוצאות .

13. **הצגת תוצאות** –יש להעדיף גרפים ותצוגות ברורות על פני טבלאות ארוכות וצפופות, כאשר הדבר מתאים .

14. **הסבר תוצאות** –לכל גרף יש לצרף הסבר קצר: מה מוצג בו, מה משמעות הצירים, ומה המסקנה העיקרית שניתן להסיק ממנו .

15. **איכות הגרפים** –גרפים צריכים לכלול כותרת, שמות צירים, ומקרא כאשר יש יותר מסדרה אחת.

בהצלחה!

שי בושינסקי