# סילבוס
# שם המחלקה:

| Course Title | | חומרה מאיצה | שם הקורס |
|---|---|---|---|
| .Course No | 3624550-20 | | מספר קורס |
| Lecturers' Name | | אורן גנון | מרצי הקורס |
| Year | 2025-2026 | תשפ"ו | שנת הוראה |
| Weekly Hours | 2שעות הרצאה + 2שעות תרגול | | היקף הקורס בש״ש |
| Credits | 3 | | נקודות זכות |
| Prerequisite | מערכות סיפרתיות<br>תכן של חומרת מחשבים | | דרישות קדם |

## תקציר הקורס

מטרת הקורס היא לחשוף את הסטודנטים לעולם **המאיצים החישוביים המקביליים**, ובפרט **GPU – Graphics Processing Unit** כמאיץ מקבילי רב־עוצמה. במהלך הקורס הסטודנטים ירכשו ידע תיאורטי ומעשי על עקרונות הארכיטקטורה של GPUs, המודלים של עיבוד מקבילי והאינטראקציה שלהם עם מערכת ההפעלה והמעבד המרכזי (CPU).

הקורס שם דגש על **Hands-On Programming**: הסטודנטים יכתבו קוד ב־CUDA כדי ליישם בפועל את העקרונות הארכיטקטוניים שנלמדים בכיתה, יתנסו בניהול משאבים על GPU, יבצעו אופטימיזציות לביצועים, ולבסוף יפתחו מערכת מואצת שמשלבת תאוריה עם יישום חישובי אמיתי.

## Course Summary

The goal of this course is to introduce students to the world of **parallel computing accelerators**, with a special focus on the **GPU – Graphics Processing Unit** as a powerful parallel accelerator. Students will gain both theoretical and practical knowledge about GPU architectures, parallel processing models, and their interaction with the operating system and the CPU.

The course emphasizes **Hands-On Programming**: students will write CUDA code to implement the architectural concepts learned in class, practice GPU resource

management, apply performance optimizations, and ultimately develop an accelerated system that combines theoretical foundations with real-world computational applications.

| | |
|---|---|
| אופני הוראה עיקריים | **שיעורי הרצאות פרונטליות** יחד עם סדנה ותרגול מעבדה<br>הקורס הוא היברידי כלומר בחלקו מקוון בחלקו פנים-אל-פנים / |

## תוצאות הלמידה

הגדרת תוצאות למידה: מה שהסטודנטיות.ים ידעו או יוכלו לעשות בסיום הקורס

**הבנה תאורטית**
- להסביר את העקרונות המרכזיים של עיבוד מקבילי והבדלים בין CPU ל־GPU.
- לתאר את מבנה הארכיטקטורה של GPU (כולל SMs, CUDA cores, היררכיית זיכרון).
- להבין את תפקיד מערכת ההפעלה ותשתיות התוכנה בתמיכה במאיצים מקביליים.

**כישורים טכניים**
- לכתוב קוד בסיסי ומתקדם ב־CUDA תוך שימוש נכון ב־Threads, Blocks ו־Grids.
- לנהל ולהעביר נתונים בין CPU ל־GPU ולבצע שימוש יעיל בזיכרון משותף וגלובלי.
- לבצע פרופיילינג ואופטימיזציה לקוד GPU כדי לשפר ביצועים (memory coalescing, occupancy optimization).

**יישום מעשי - בתרגולים**
- ליישם אלגוריתמים חישוביים נפוצים (כגון כפל מטריצות, Reduction, Prefix Sum) בסביבת CUDA.
- לפתח מערכת מואצת מבוססת GPU המשלבת ידע תאורטי עם מימוש פרקטי.
- להעריך ביצועים ולנתח trade-offs בין פתרון סדרתי לפתרון מקבילי.

**תוצאות למידה – מיני־פרויקט**

בסיום המיני־פרויקט הסטודנטיות והסטודנטים יוכלו:

1. **יישום מעשי של עקרונות CUDA**
   - ממש אלגוריתם חישובי אמיתי על גבי GPU באמצעות CUDA.
   - לבצע חלוקה נכונה של Threads, Blocks ו־Grids בהתאם לבעיה.
2. **ניהול משאבים וזיכרון**
   - לנהל ביעילות העברת נתונים בין CPU ל־GPU.
   - להשתמש במבני זיכרון שונים (Shared, Global, Constant) כדי לשפר ביצועים.
3. **ביצוע אופטימיזציות**
   - לזהות צווארי בקבוק בביצועים באמצעות כלי פרופיילינג.
   - ליישם שיטות אופטימיזציה (כמו Memory Coalescing או שימוש ב־Shared Memory) לשיפור ריצה.
4. **מיומנויות חקר וחשיבה ביקורתית**
   - להעריך Trade-offs בין פתרון סדרתי לפתרון מקבילי.
   - להשוות בין מימוש נאיבי למימוש מואץ ולנתח שיקולים של דיוק מול ביצועים.
5. **עבודת צוות והצגה**
   - לעבוד בצוות קטן לפתרון בעיה חישובית מקבילית.
   - להציג את הפרויקט, התוצאות והמדדים (ביצועים, דיוק, צריכת זיכרון) בצורה מסודרת ושקופה

| | |
|---|---|
| | |
| | |
| | |

| מבנה הערכה | | |
|---|---|---|
| קטגוריה | % מהציון | הערות |
| פרויקט גמר | %70 | |
| תרגילי להגשה | %30 | |
| | %__ | |
| | %__ | |
| סה"כ | 100% | |

| | |
|---|---|
| חובות הסטודנטים | על פי תקנון שנקר. |

| מבנה הקורס | | | |
|---|---|---|---|
| | מספר/תאריך | נושא השיעור | פירוט |
| | 1 | מבוא למאיצים חישוביים | נלמד על ההבדלים בין CPU ל־GPU, הרקע לצורך במקביליות, חוק אמדל והשפעתו על ביצועים. בתרגול: חישובים סדרתיים מול מקביליים, הדגמה בסיסית של האצת ביצועים. |
| | 2 | מאיצים מוכווני תפוקה (Throughput-oriented Accelerators) | את עקרונות התכנון של GPU כמאיץ מקבילי חב־תפוקה. בתרגול: כתיבת תוכנית CUDA ראשונה, הרצת Kernel פשוט. |
| | 3 | אופטימיזציות ב־GPU | זאה: שיטות לניצול מקסימלי של ליבות GPU, ול latency על ידי SIMT scheduling. תרגול: זודה עם Nsight / Profiler – ניתוח ביצועים בסיסי (self-study מודרך). |
| | 4 | מודלי זיכרון ב־GPU | |

| | | | |
|---|---|---|---|
| | | | רצאה: היררכיית הזיכרון (גלובלי, משותף, קומי, Registers). תרגול: מימוש אלגוריתם CU (חלק 1) תוך שימוש נכון במבני הזיכרון. |
| | 5 | מודל התכנות ו־API של CUDA | אה: ארגון Thread-Block-Grid, שימוש ב־API של CUDA לניהול משאבים. תרגול: מימוש גוריתמים נוספים (חלק 2) – למשל Vector Add או Reduction. |
| | 6 | זיכרון ולוקליות של נתונים | רצאה: חשיבות locality, caching ושימוש ב־Shared Memory. תרגול: שימוש ב־cuda::atomic לפתרון מצבים של גישה מתחרה לנתונים. |
| | 7 | תבניות חישוב מקבילי: Convolution ושיקולי נקודה צפה | רצאה: מימוש קונבולוציה על GPU, אתגרי Floating-P, דיוק מול ביצועים. תרגול: מימוש ל מטריצות ב־CUDA תוך שימוש ב־Shared Memory. |
| | 8 | הרצאת העשרה: ארכיטקטורות מתקדמות | אה: היכרות עם מאיצים נוספים (TPU, DPU, AI Accelerato), מגמות עכשוויות בתחום. ל: העמקה חופשית בנושאי CUDA שנבחרים ע״י הסטודנטים. |
| | 9 | תבניות חישוב – Prefix Sum (Parallel Scan) | אה: אלגוריתם Prefix Sum במימוש מקבילי. גול: מימוש האלגוריתם ב־CUDA תוך ניצול זיכרון משותף. |
| | 10 | חישוב היסטוגרמה מקבילי | רצאה: בעיות scaling, contention, ושיטות יעול חישוב היסטוגרמות על GPU. תרגול: מימוש היסטוגרמה מקבילית עם CUDA. |

| | | | |
|---|---|---|---|
| | | | |
| | 11 | Sparse Matrix Computation | אה: אתגרים של חישוב על מטריצות דלילות, צוגים שונים (CSR, COO). תרגול: התחלת י־פרויקט – בחירת בעיה חישובית ויישום ראשוני ב־CUDA. |
| | 12 | Threading Hardware ומבנים מתקדמים | Threadingהרצאה: סקירת חומרת ה־ Streams נושאים מתקדמים (כמו ב־GPU, וEvents־). תרגול: המשך וסיום **מיני־פרויקט** והצגת תוצאות. |
| | 13 | GPU + CPU Heterogeneous Programming<br>● | הרצאה: שילוב עבודה היברידית בין CPU ל־GPU, מודלים של חלוקת עומס, השוואה בין ביצועים ותיאום משאבים. היכרות עם ספריות כמו OpenACC או Thrust. |
| | 14 | Deep Learning on GPUs<br>● | הרצאה: סקירת היישומים המרכזיים של GPU בלמידה עמוקה – אימון רשתות נוירונים, פעולות בסיסיות (Convolution, Backpropagation). תרגול: יישום Forward Pass קטן ברשת נוירונים ב־CUDA, או שימוש בספרייה קיימת (cuDNN) כדי להריץ רשת פשוטה. |

| רשימה ביבליוגרפית | |
|---|---|
| רשות | : "Programming Massively Parallel Processors: A Hands-on Approach" |
| רשות | University of Oxford: CUDA Programming on NVIDIA GPUs Taught by Mike Giles, Professor |
| רשות | UC Davis: EE171: Parallel Computer Architecture Taught by John Owens, Associate Professor |

| | |
|---|---|
| University of Sheffield: COM4521: Parallel Computing with GPUs Taught by Paul Richmon | רשות |