# מוסכמות כתיבה הנהוגות בקורס

בשפת ג'אווה נהוג להשתמש במוסכמות (קונבנציות) מסוימות בנוגע לכתיבת קוד בשפה. מוסכמות אלו אינן חלק מהגדרת השפה (כלומר, אי עמידה בהן לא תגרור שגיאות קומפילציה) אולם חשיבותן רבה – כתיבת קוד תוך הצמדות למוסכמות משפרת את קריאות הקוד ומקלה על הבנתו, ולכן בקורס זה אנו נקפיד על כתיבה במוסכמות הנהוגות.

להלן המוסכמות:

**כללי כתיבת שמות**

1. **שמות מחלקות** יתחילו תמיד באות גדולה. שאר השם יכיל אותיות קטנות או מספרים. למשל: House, Student. אם יש יותר ממילה אחת בשם המחלקה, כל מילה נוספת תתחיל גם באות גדולה, למשל: ComputerClass, SpaceshipToTheMoon.
2. **שמות משתנים** יתחילו תמיד באות קטנה. שאר השם יכיל אותיות קטנות או מספרים. למשל: name, grade. אם יש יותר ממילה אחת בשם המשתנה, כל מילה נוספת מלבד המילה הראשונה תתחיל באות גדולה. למשל: studentGrade, maxNumberOfDrivers, point1And2.
3. **שמות של תכונות (attributes)** יתחילו תמיד בקו תחתון שאחריו אות קטנה. למשל: _name, _id, _x. אם יש יותר ממילה אחת בשם התכונה, כל מילה נוספת חוץ מהמילה הראשונה תתחיל באות גדולה. למשל: _xCoordinate, _bookTitle.
4. **שמות של קבועים** יכילו אך ורק אותיות גדולות. למשל: DELAY, PI. אם יש יותר ממילה אחת בשם הקבוע, המילים יופרדו אחת מהשניה בקו תחתון, למשל: FRAME_WIDTH, MAX_ALLOWED_SPEED.
5. **שמות של שיטות** יתחילו באות קטנה. אם שם השיטה מכיל יותר מאות אחת, כל מילה נוספת חוץ מהראשונה תתחיל באות גדולה. למשל: calculatePrice, getUserName.
6. **שמות של שיטות למניפולציה של תכונות פרטיות** – שיטות ציבוריות שאמורות לתת למשתמש גישה לתכונות פרטיות של המחלקה יתחילו במילים get או set שאחריהן שם התכונה. שיטת get נועדה **להחזיר** את ערך התכונה, שיטת set נועדה **לשנות** את ערך התכונה. למשל, בהינתן התכונות _studentName, _x, נכתוב את השיטות הבאות: getX, setX, getStudentName, setStudentName.

**כללי כתיבה והרגלים מועילים אחרים**

ההרגלים הבאים יעזרו לכם לכתוב קוד ברור וקריא יותר, ועובדה זו תעזור לכם לאתר טעויות בקוד וגם להימנע מטעויות בכתיבה.

1. **הזחה (indentation)** – כל בלוק של קוד בתוכנית יתחיל ביישור של מספר רווחים ימינה. כדאי לעשות הרגל וכל פעם כשפותחים סוגריים מסולסלים, עושים `enter` ואז `tab`. כך ברור בעיניים למי הקוד שייך ובאיזו רמת קינון הוא נמצא.

2. **שורות חדשות** – נא לא לחסוך בשורות חדשות. כדאי להפריד קטעי קוד שלא מבצעים את אותה משימה בשורה חדשה, ולא לכתוב את כל הקוד בבלוק אחד גדול של שורות.

3. **ריווח** – כדאי להשתמש ברווחים איפה שאפשר (לרוב, שפת ג'אווה עיוורת לרווחים וזה לא משנה לה אם שמתם רווח או לא. לבן האדם זה משנה מאוד). הרגל שכדאי לאמץ – לפני ואחרי כל פעולה חשבונית או אופרטור לוגי שימו רווח. למשל:

```
x = y + z * 5;
if( num == 8 )
for( int i = 0 ; i < 10 ; i++)
System.out.println("My name is: " + name + ".");
```

4. **מיקום הפקודות** – כל פקודה תיכתב בשורה משלה. אנא הימנעו מקוד שנראה כך:

```
int x; double d; x=2; d=3.1; System.out.print(x+","+d);
```

זה לא נראה טוב, זה מקשה על הקריאה ויקשה בעתיד על איתור שגיאות.

5. **הערות ותיעוד** – תיעוד התוכנית נועד להסביר למי שיקרא אותה מה משמעות הפקודות ומה התוכנית עושה. התיעוד נועד גם למתכנת שכתב את הקוד בעצמו, כדי שיוכל להיזכר לאחר זמן במשמעות הקוד שכתב. אין כללים מוחלטים לכתיבת תיעוד, אבל הנה כמה כללי אצבע – בראש כל קובץ נהוג לכתוב הערה שמכילה את שם המחלקה, שם כותב המחלקה ותאריך הכתיבה (או העדכון האחרון). בנוסף ההערה תכיל תיאור קצר ותמציתי שמסביר מה תפקיד המחלקה ומה היא עושה. **לפני כל שיטה** נהוג לכתוב הערה שמסבירה בקצרה מה מבצעת השיטה, אילו פרמטרים היא מקבלת ואת משמעותם, ומה היא מחזירה (אם בכלל). **לפני כל קטע קוד לא טריוויאלי** נהוג לכתוב הערה שמסבירה מה קטע הקוד מבצע, ואם צריך גם את הדרך שבה הוא עושה זאת. **אין צורך** לעומת זאת, לכתוב חיבור בן 1000 מילים שמתאר את התפתחות התוכנה עד היום. תיאור תמציתי וקולע עדיף בהרבה וגם יהיה יעיל יותר וקל יותר לקריאה והבנה.

6. **שמות משמעותיים** - שמות המחלקות, המשתנים, התכונות והשיטות צריכים להיות משמעותיים ככל שניתן. הכוונה היא שהשם כבר "יגיד" מה תפקידו של המשתנה או השיטה. למשל, אם ישנה תכונה ששומרת את שם הסטודנט, כדאי לקרוא לה `_studentName` ולא סתם `_n`. אם יש שיטה שמוצאת את הציון המקסימלי כדאי לקרוא לה `findMaxGrade` ולא סתם `f`. מצד שני, שוב, לא צריך להגזים ולתת שמות

משמעותיים בלבד. למשל, אם יש שיטה שמוצאת את שטחו של המלבן הכי גדול מבין אוסף מלבנים, עדיף שנקרא לה `findMaxRecArea` ולא בשם המשמעותי מאוד: `runOverEachRectangleAndFindTheBiggestArea`.