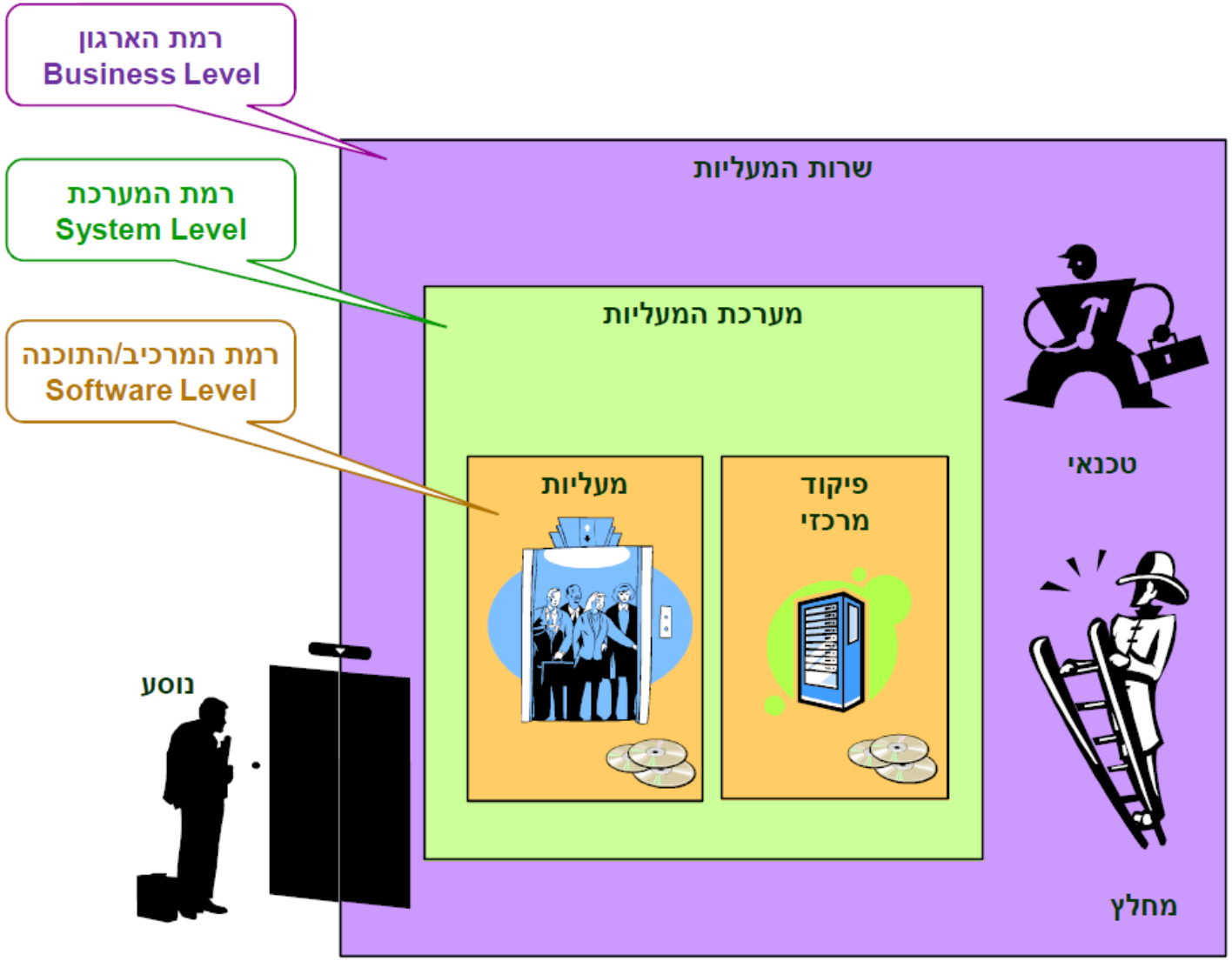
מבוא להנדסת תכנה

מרצות: נעמי אונקלוס-שפיגל
מורן קופפר

תוכן ההרצאה

- מבוא
- מוטיבציה
- מרכיבי הדיאגרמה
- דוגמא

שרות המעליות



רמת המרכיב/התוכנה

- מטרת התוכנה – לממש את המערכת
- מערכת המעליות ממומשת באמצעות מרכיבים שונים, בתוך התוכנה יש ארגון של המבנה ותיאור הביצוע
  - מבנה: המידע והפונקציונאליות כמוסים בתוך "עצמים"
  - ביצוע: העברת "הודעות" בין אובייקטים
- ברמת התוכנה יש להציג את מונחי מרחב הבעיה והקשרים ביניהם

שרות המעליות – רמת התוכנה

קומה
- מספר
- ?

מעלית
- זיהוי
- מצב: {פעילה, מושבתת}
- תקינות: {תקינה, מקולקלת}
- מיקום: קומה
- תכנית: תור של בקשות

- נסיעה (כיוון)
- עצירה

כפתור
- דולק: {כן, לא}
- זיהוי: קומה
- כיוון: {למעלה, למטה}

- לחיצה
- הדלקה/כיבוי

רלוונטי לכפתור-מעלית בלבד

רלוונטי לכפתור-קומה בלבד

נוסע
- ?
- ?

דלת
- שיוך: מעלית
- מצב: {פתוחה, סגורה}

- פתיחה/סגירה

בקשת עצירה
- שיוך: מעלית
- יעד: קומה

- פתיחה
- ביטול

דיאגרמת CLASS

- הלב של UML
- דיאגרמה מבנית סטטית המתארת את מבנה המערכת
- כוללת תיאור ויזואלי של המחלקות במערכת והקשרים ביניהן

דיאגרמת CLASS

- המחלקות שבמודל המחלקות אמורות לספק את כל הפונקציונליות המערכתית
- מכל דרישה פונקציונלית יש להצביע למחלקה או למחלקות הרלוונטיות
  - משתתפות בדרישה תפעולית (OR)
    - לדוגמה: "אם לא היה דלוק קודם נדלק הכפתור בעקבות הלחיצה" ← כפתור
  - מספקות את מבני הנתונים עבור דרישות המידע (DR)
    - לדוגמה: "בכל קומה יהיו שני כפתורים" ← קומה
- מכל מחלקה ב- Class יש להצביע על הדרישות הפונקציונליות הרלוונטיות לה

מועמדים לעצמים

- עצמים המייצגים ישויות פיזיות (מנוע, דלת, עמדת עבודה, ...)
  - תכונות: פרמטרים ונתונים לגבי הישות, קלט/פלט
  - אופרטורים: פונקציונלית פיזית
    - העצם המייצג משמש, למעשה, כממשק שבית התוכנה לישות הפיזית
- עצמים המייצגים ישויות לוגיות (תהליך, שירות, ...)
  - תכונות: פרמטרים ונתונים לגבי הישות, קלט/פלט
  - אופרטורים: פעולות המשמשות את התהליך/השירות
- עצמים המייצגים ישויות מידע (מאגרי נתונים, רשימות, תורים, ...)
  - תכונות: רכיבי המידע שבאחריות הישות
  - אופרטורים: פעולות על המידע (אחסון, שליפה, עדכון, ...)
- עצמים המייצגים עצמים הנמצאים ברכיב תוכנה אחר
  - "שיקוף" של העצמים החיצוניים
  - מימוש ממשקי תוכנה-תוכנה דרך תווך של חומרה

תוכן ההרצאה

- מבוא
- מוטיבציה
- מרכיבי הדיאגרמה
- דוגמא

למה צריך CLASS?

- המטרה הינה הבהרה וחידוד של המונחים והיחסים ביניהם
- שימושים:
  - יישוב סתירות ואי-בהירויות במפרטי הלקוח
  - מילון מונחים של המערכת
  - בסיס לישויות המידע בהן נדרשת התוכנה לטפל

תוכן ההרצאה

- מבוא
- מוטיבציה
- מרכיבי הדיאגרמה
- דוגמא

מחלקות

- אוסף של אובייקטים אשר יש להם את אותם מאפיינים
- מחלקה מסומנת על ריבוע המחולק ל- 3 חלקים
  - טיפוס: שם המחלקה
  - תכונות: שם של מאפיין של מחלקה אשר מייצג תחום של ערכים שמופע יכול לקבל
  - אופרטורים: מימוש של שירותים שניתן לבקש מהמחלקה

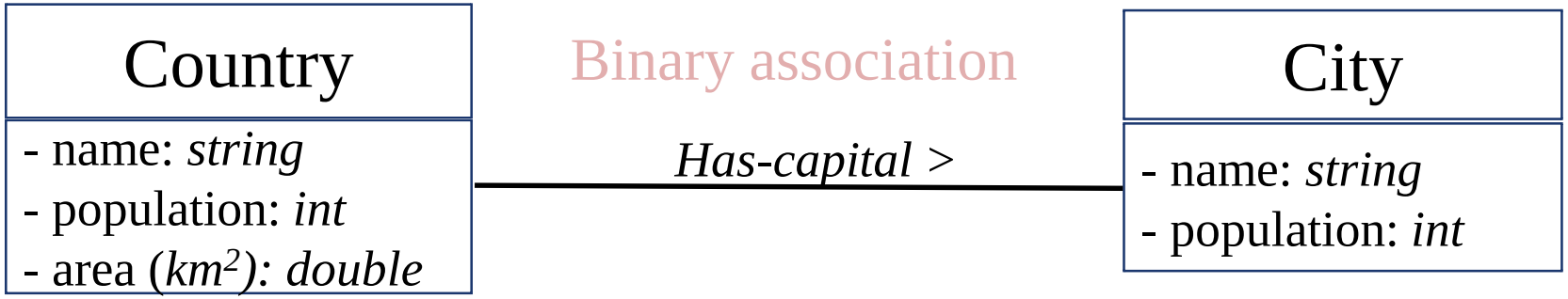
Class_Name
- attribute : type
+ operation() : type + operation(parameters) : type

יחסים בין מחלקות

- יחס הינו חיבור בין מחלקות.
- יחסים נפוצים בין מחלקות:
  - זיקה (Association)
  - הקבצה (Aggregation)
  - הרכבה (Composition)
  - הכללה - הורשה (Generalization)

יחס זיקה (ASSOCIATION)

- יחס זיקה הינו יחס מבני אשר מציין שאובייקטים של מחלקה מסוימת מחוברים לאובייקטים של מחלקה אחרת
- יחס זיקה מסומן ע"י קו בין שתי מחלקות.
  - ניתן לבצע קשר זיקה עצמי ("שכפול" של מחלקה)
  - ניתן לתת שם לקשר כדי לתאר את טבע היחסים בין המחלקות



ריבויים

- קשר מציין שאחד (לפחות) מהמחלקות המקושרות מצביע למחלקה השנייה בקשר.
- נעשה שימוש בריבויים כדי לציין כמה אובייקטים ניתן לחבר באמצעות מופע של קשר.
- כאשר מספר x מופיע בסוף הקשר הוא מציין שכל אובייקט של המחלקה מהצד ההופכי צריכה להיות מחוברת לכמות של x אובייקטים של המחלקה

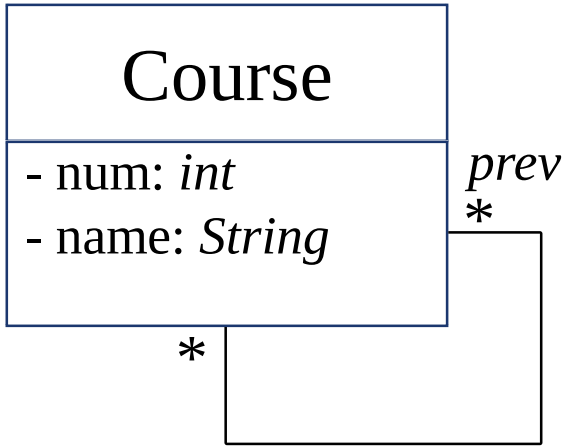
0..1 – עד אחד
1 – בדיוק אחד
* - אפס עד רבים
1..* - לפחות אחד
x – כמות x
0..x – אפס עד כמות x
x..y – x עד כמות y

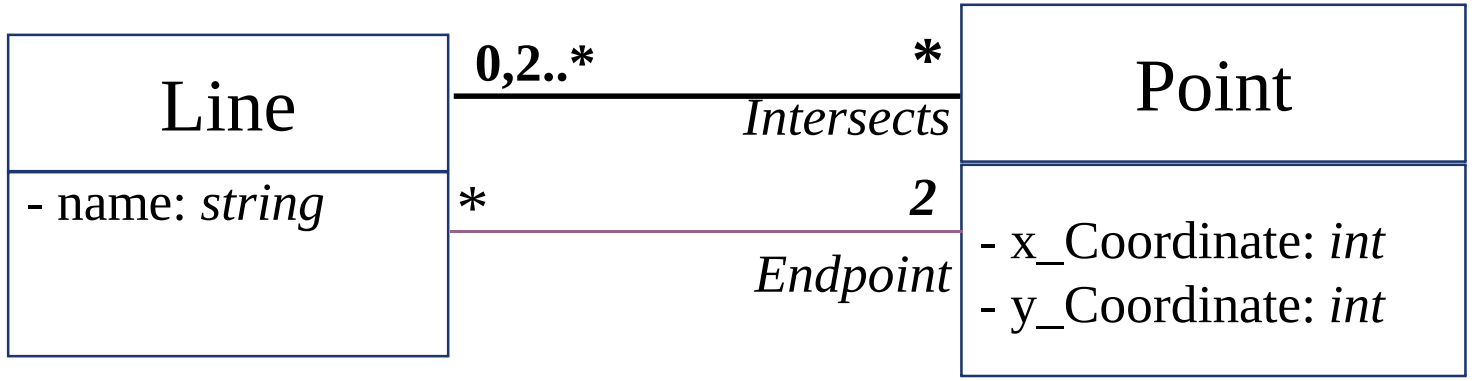
דוגמא לריבויים

- במחלקה יש לפחות עובד אחד
- עובד שייך למחלקה אחת בדיוק
- לעובד יכול להיות 0-1 משרדים, כלומר, לעובד יכול להיות משרד או שלא
- בכל משרד יש בין 1-10 עובדים

תפקיד בקשר (ROLE NAME)

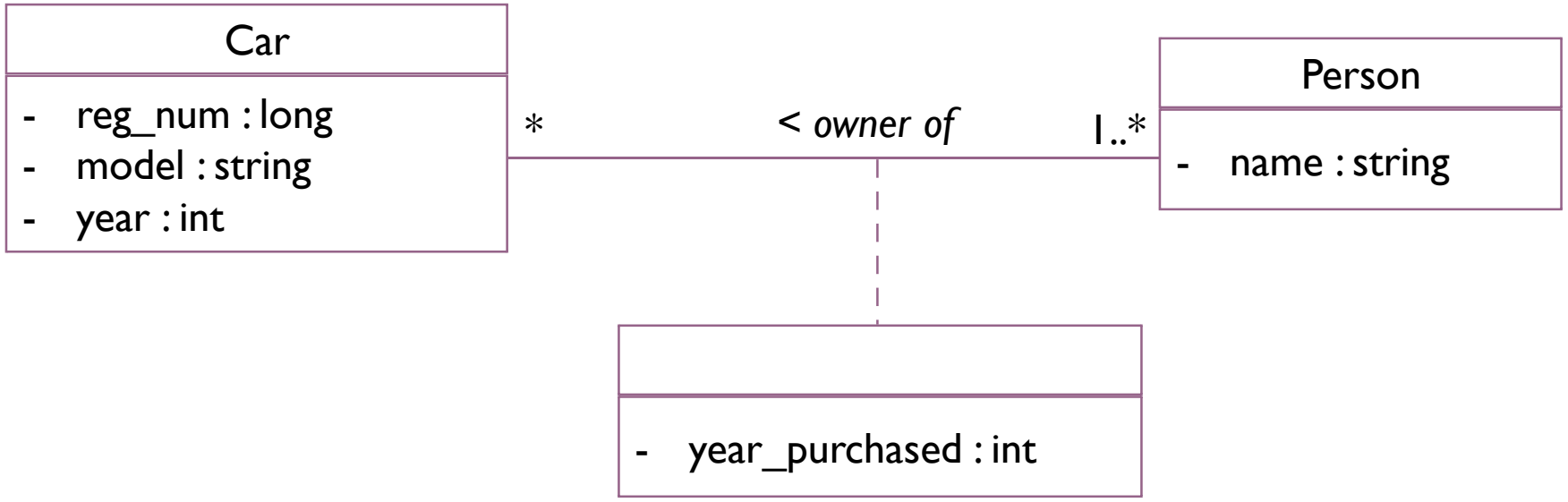
- ניתן להוסיף לכל מחלקה תפקיד על קשר
  - שם תפקיד המחלקה בקשר יופיע בסוף הקשר
- תפקיד המחלקה מועיל כאשר:
  - יש יותר מקשר אחד בין המחלקות
  - הקשר הינו קשר עצמי



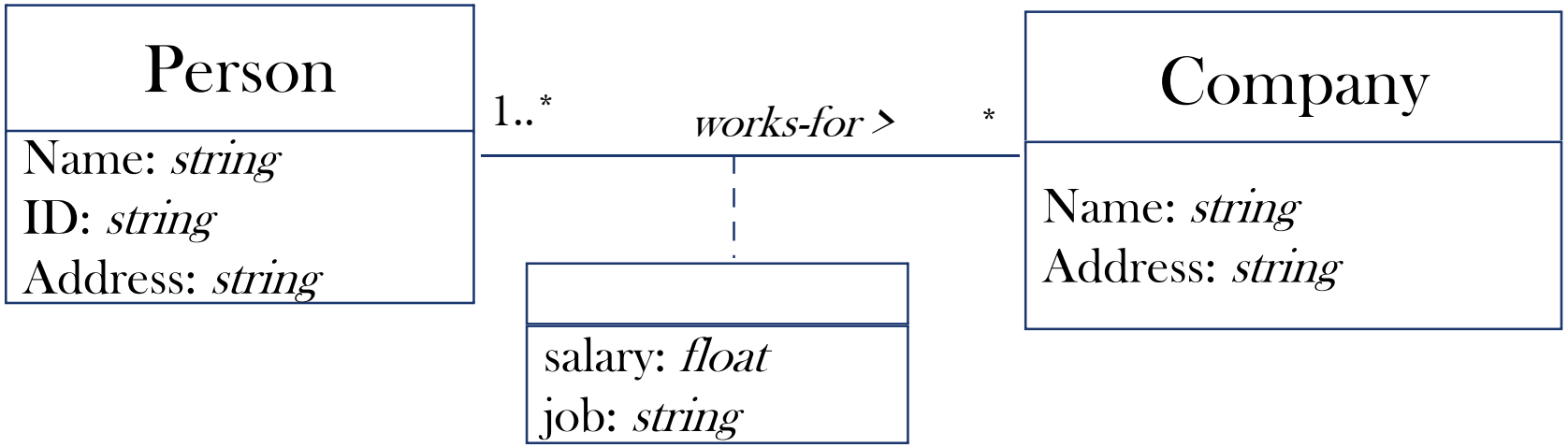


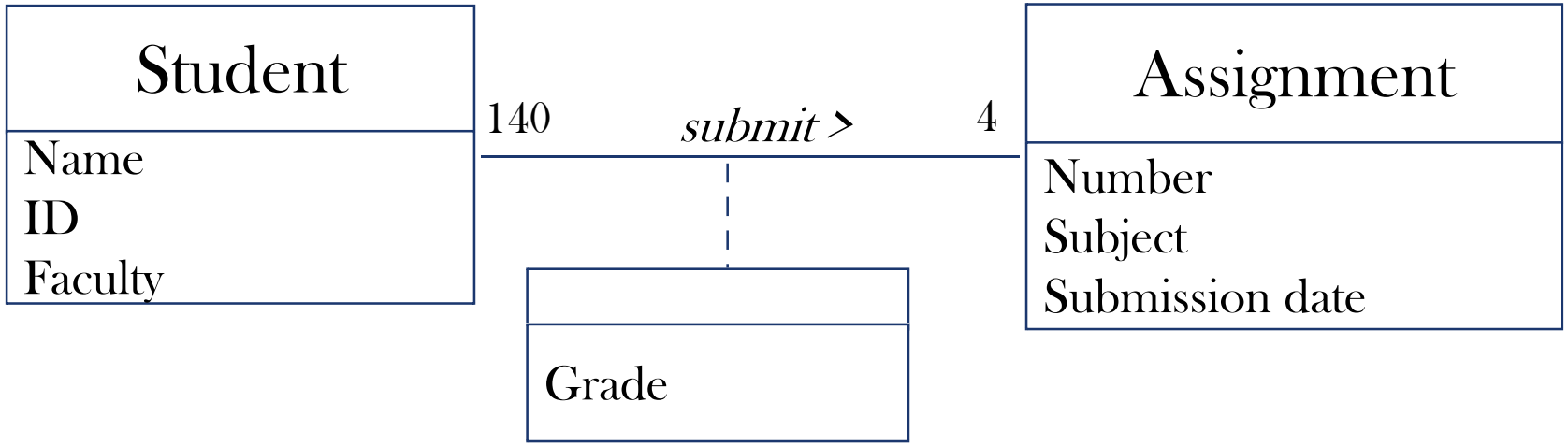
מחלקת קשר

- מחלקה המקושרת ליחס זיקה
- מייצגת מאפיינים של הקשר אשר אינם שייכים לאף אחת מהמחלקות בקשר



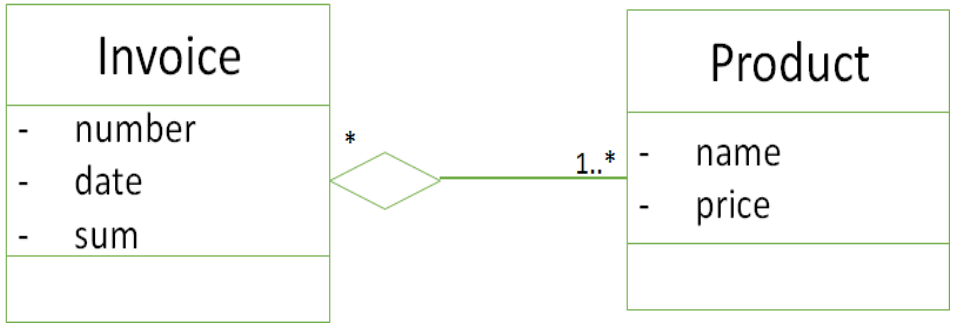
מחלקת קשר - דוגמאות





הקבצה (AGGREGATION)

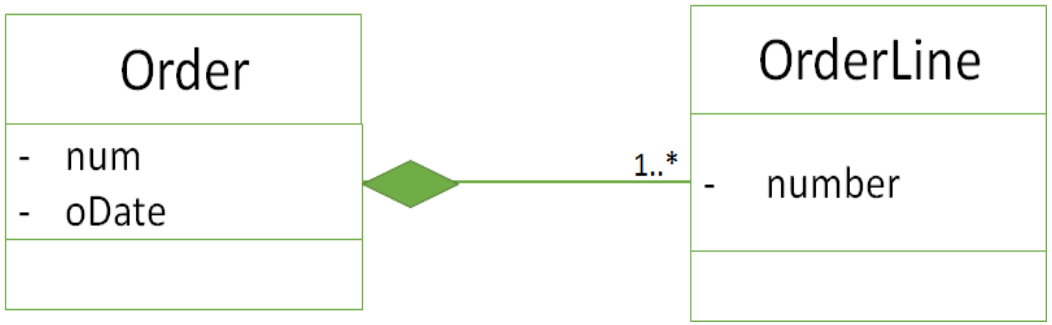
- קשר של שלם וחלקיו
  - דוגמא: מחלקה invoice מכילה אובייקטים של Product



- בקשר הקבצה ל- "חלק" יש זכות קיום גם ללא השלם
  - בדוגמא: יכול להיות אובייקט מוצר שאיננו חלק מאובייקט חשבונית.
- ניתן להחליף קשר הקבצה בקשר זיקה רגיל

הרכבה (COMPOSITION)

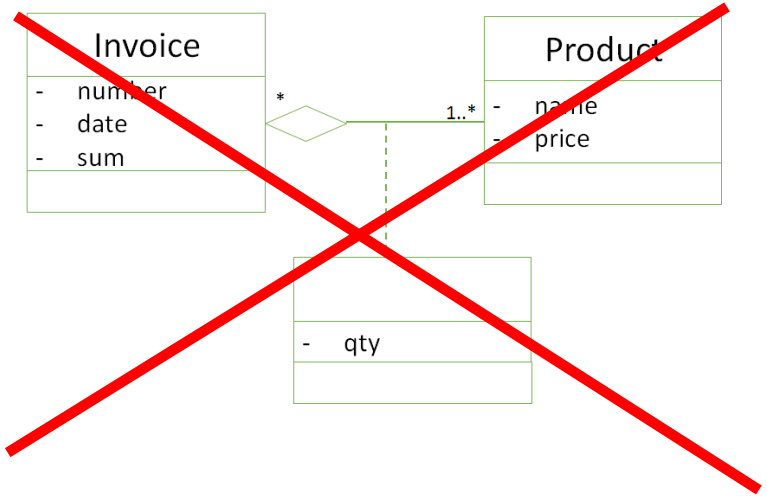
- קשר הרכבה בדומה להקבצה גם הוא מתאר שלם וחלקיו
  - דוגמא:

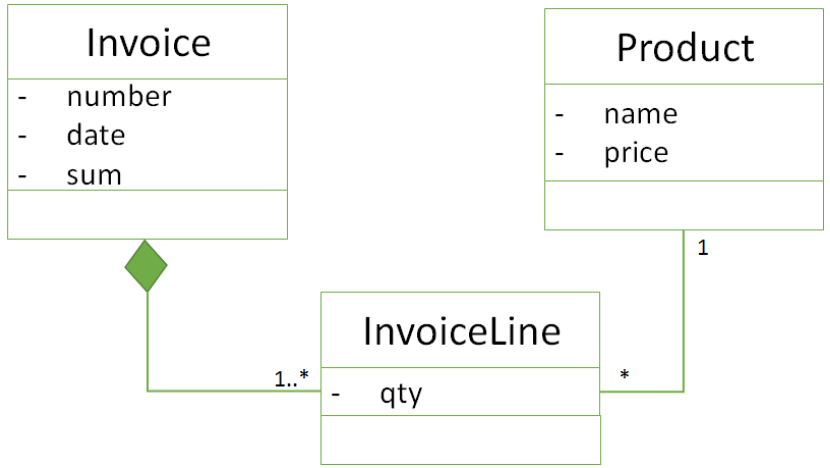


- בקשר הרכבה ל- "חלק" אין זכות קיום ללא השלם.
  - בדוגמא: לא יכול להיות אובייקט "שורת הזמנה" שאיננו חלק מאובייקט הזמנה
- קשר הרכבה **לא** ניתן להחליף בקשר זיקה

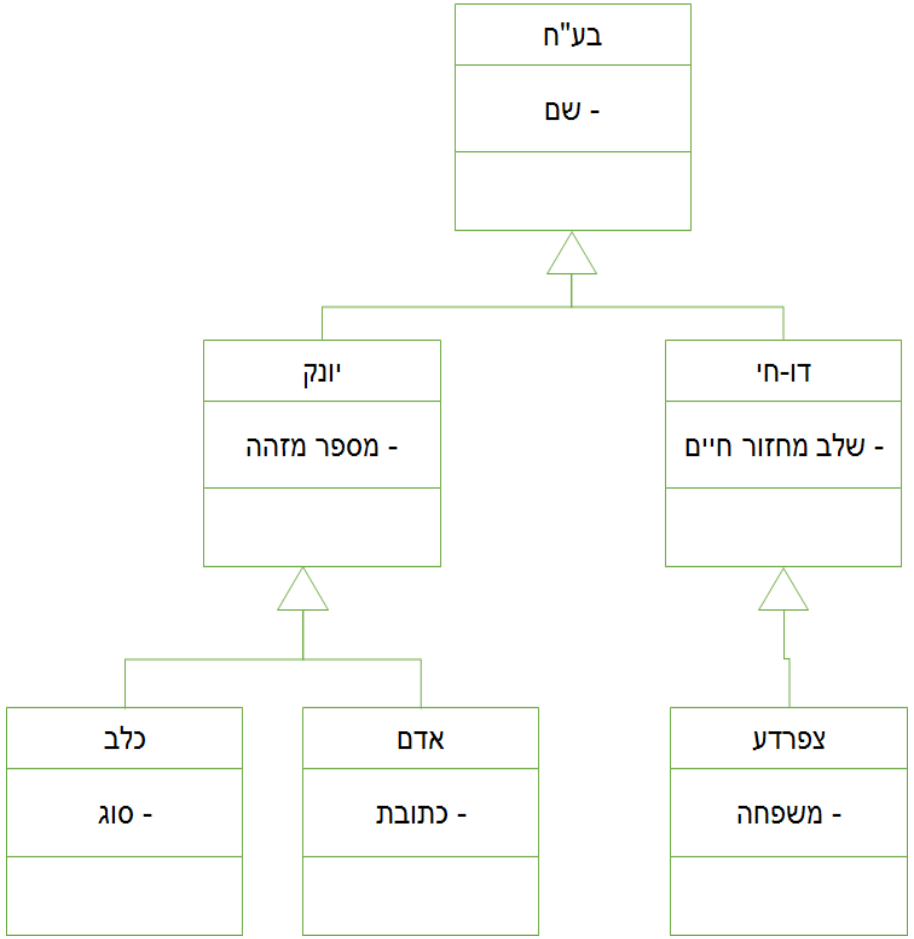
הקבצה ומחלקת קשר

- לקשר הקבצה **לא** תהיה מחלקת קשר
  - אם נדרשת מחלקת קשר נשתמש בקשר הרכבה מהמחלקה המכילה למחלקת הקשר וקשר זיקה ממחלקת הקשר למחלקה המוכלת.





הורשה



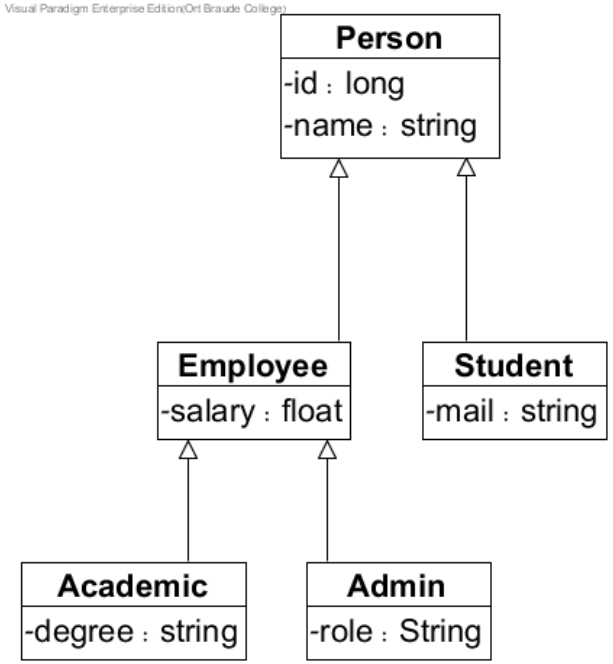
- בני האדם נוהגים לעשות קטלוג, עבודה עם היררכיה.
  - למשל אם נסתכל על כלב, אז כלב הוא יונק, ויונק הוא בע"ח
  - אם נסתכל על צפרדע, אז צפרדע הוא דו-חי, ודו-חי הוא בע"ח
- הורשה היא בעצם הגדרה של היררכיה בצורת מחלקות, אב ובן, מחלקה ותת-מחלקה, מחלקה ומחלקה נגזרת

סוגי הורשה

- סוגי הורשה:
  - complete מול incomplete
  - disjoint מול overlap
- ישנן ארבע אפשרויות לסוגי הורשה :
  - Complete disjoint
  - Complete overlap
  - Incomplete disjoint
  - Incomplete overlap

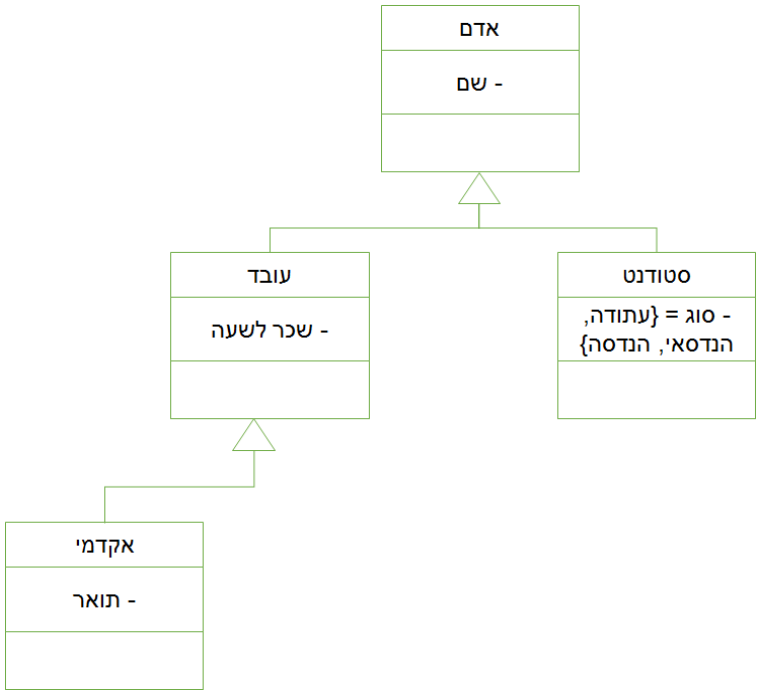
COMPLETE

- כל מחלקות הבנים מתוארות בתרשים
- דוגמת המכללה – להלן תרשים מלא:

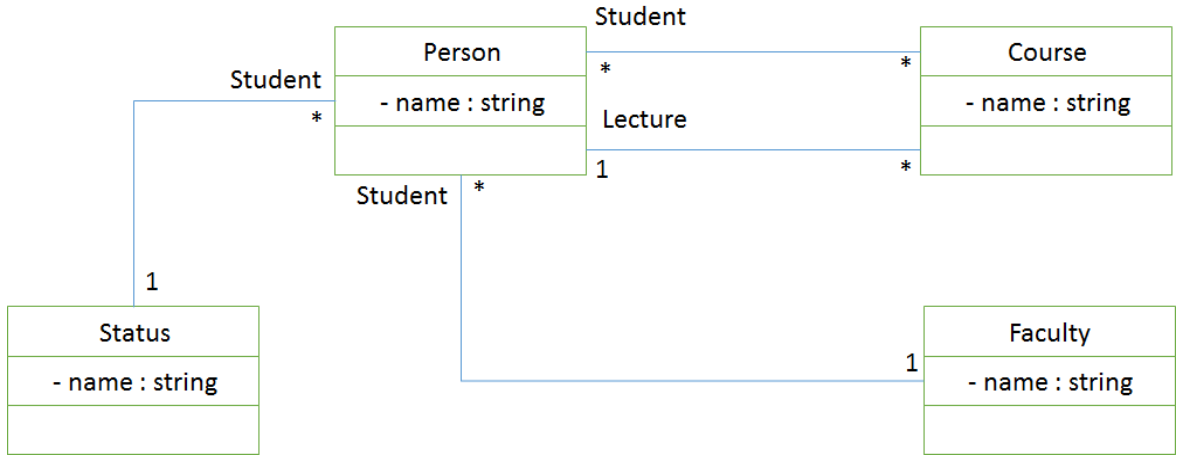


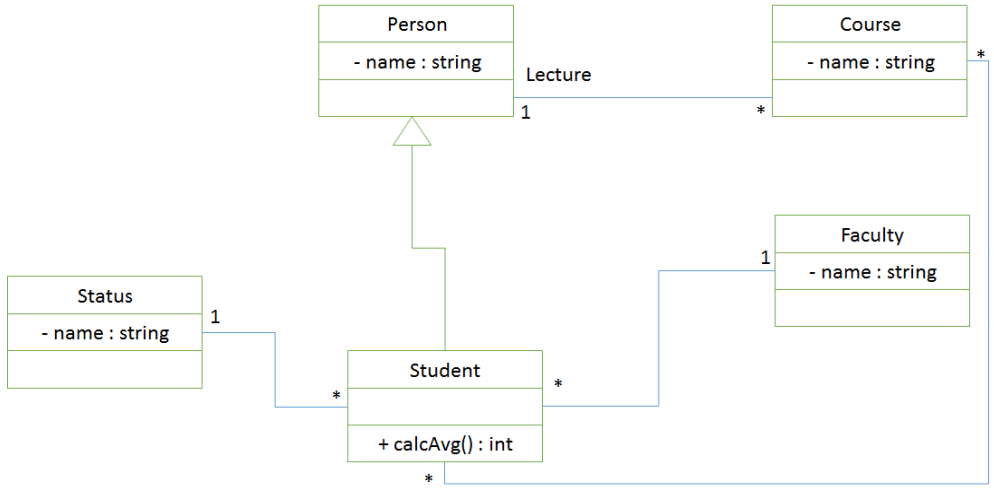
INCOMPLETE

- רק מחלקות הבנים בעלי תכונות נוספות או מתודות נוספות מתוארות בתרשים
- בדוגמת המכללה – עובד מנהלי יהיה מיוצג כאובייקט של מחלקת עובד



ROLE NAME והורשה



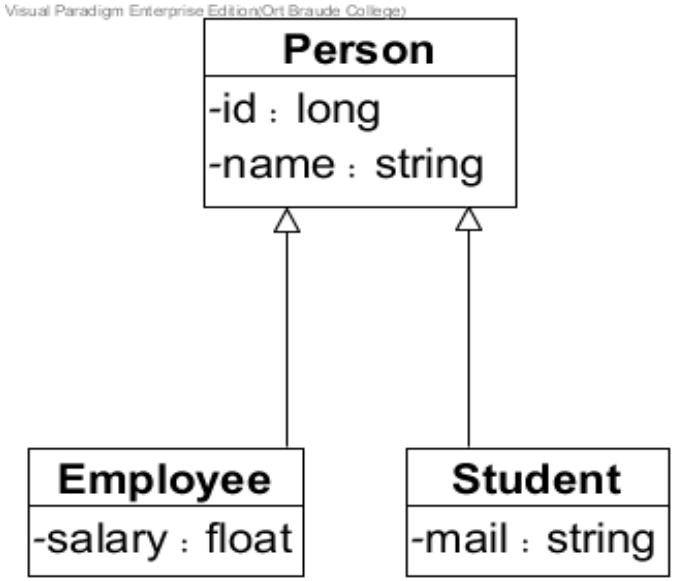


ה- role של student מופיע יותר מפעם אחת

ולכן ניתן לייצגו כמחלקת בן של person.

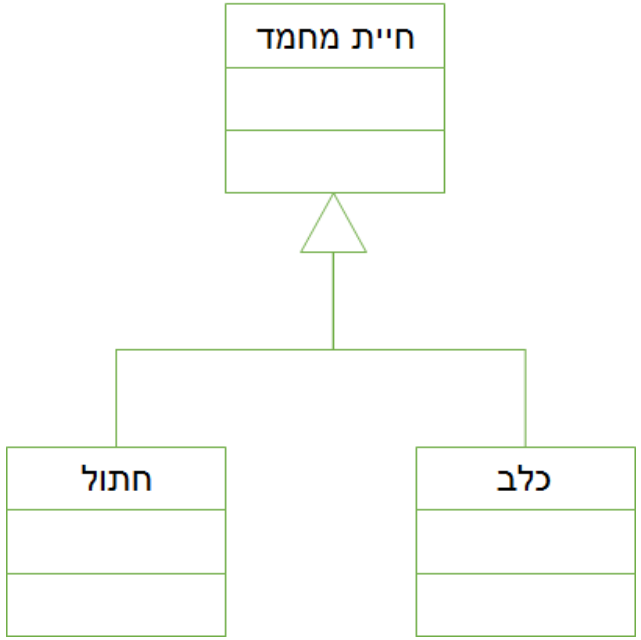
OVERLAP

- אובייקט יכול להיות שייך ליותר ממחלקת בן אחת.
  - למשל אובייקט x יכול להיות גם מסוג סטודנט וגם מסוג עובד



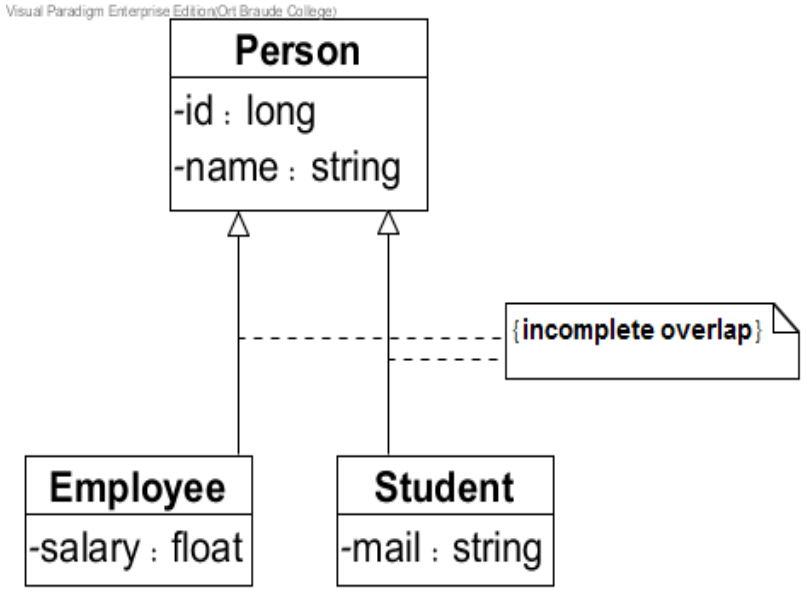
DISJOINT

- אובייקט שייך למחלקת בן אחת בלבד.
  - למשל כלב לא יכול להיות חתול ולהיפך.



משמעות סוגי ההורשה

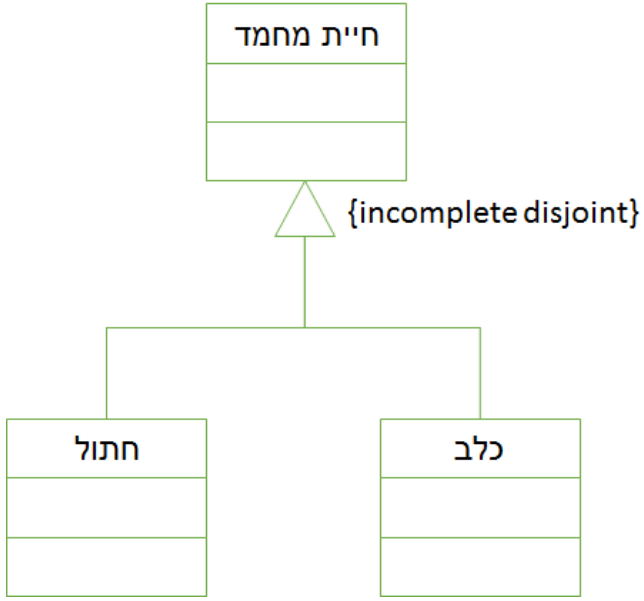
- סוגי ההורשה השונים מציינים אילו אובייקטים ניתן ליצור
  - INCOMPLETE OVERLAP



- בגלל שסוג ההורשה הוא incomplete אז ניתן ליצור אובייקטים שהם לא סטודנט ולא עובד ← ניתן ליצור אובייקטים ממחלקה "אדם"
- בגלל שהקשר הוא overlap אז אובייקט יהיה אדם ו/או סטודנט ו/או עובד

משמעות סוגי ההורשה

- סוגי ההורשה השונים מציינים אילו אובייקטים ניתן ליצור
  - INCOMPLETE DISJOINT
    - בגלל שסוג ההורשה הוא incomplete אז ניתן ליצור אובייקטים שהם לא כלב ולא חתול ← ניתן ליצור אובייקטים ממחלקה "חיית מחמד"
    - בגלל שהקשר הוא disjoint אז אובייקט יהיה או כלב, או חתול, או חיית מחמד.



משמעות סוגי ההורשה

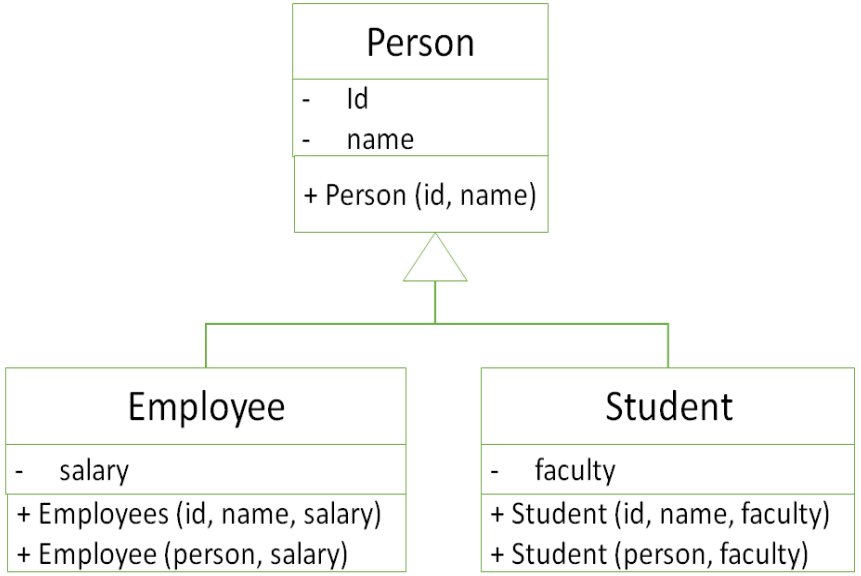
- סוגי ההורשה השונים מציינים אילו אובייקטים ניתן ליצור
  - COMPLETE OVERLAP

בגלל שהקשר הוא complete אז אין צורך ליצור אובייקט מסוג "אדם". אולם בגלל שהקשר הוא overlap אז אובייקט יכול להיות גם סטודנט וגם עובד ← ולכן כדי לא לשכפל מידע ניצור אובייקט ממחלקה "אדם"

<u>הסבר:</u>

ליצור אובייקט מסוג סטודנט נשתמש בבנאי הראשון. אם הסטודנט הנ"ל נרצה להגדירו גם כעובד ניצור אובייקט מסוג עובד ונשתמש בבנאי השני של עובד.

ליצור אובייקט מסוג עובד נשתמש בבנאי הראשון. אם העובד הנ"ל נרצה להגדירו גם כסטודנט ניצור אובייקט מסוג סטודנט ונשתמש בבנאי השני של סטודנט.

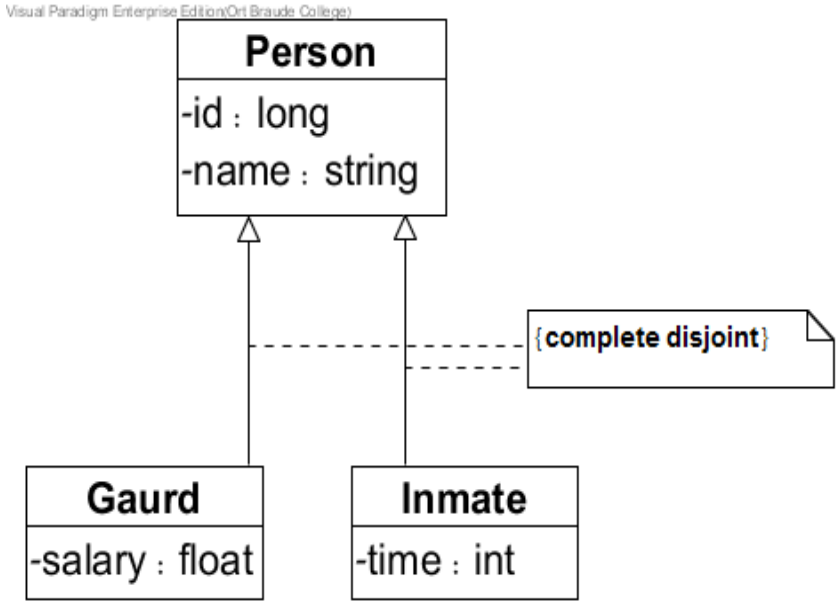


משמעות סוגי ההורשה

- סוגי ההורשה השונים מציינים אילו אובייקטים ניתן ליצור
  - COMPLETE DISJOINT

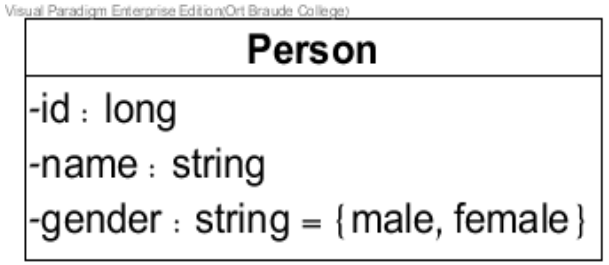
בגלל שהקשר הוא complete אז אין צורך ליצור אובייקט מסוג "אדם"
בגלל שהקשר הוא disjoint אז אובייקט יהיה או סוהר, או אסיר
← מחלקה "אדם" היא מחלקה שלא ניצור ממנה אובייקטים.
זוהי מחלקה אבסטרקטית (מופשטת)

מחלקה אבסטרקטית נסמן ע"י כתיבת שמה ב- Italic או שנכתוב ליד המחלקה אילוץ {abstract}



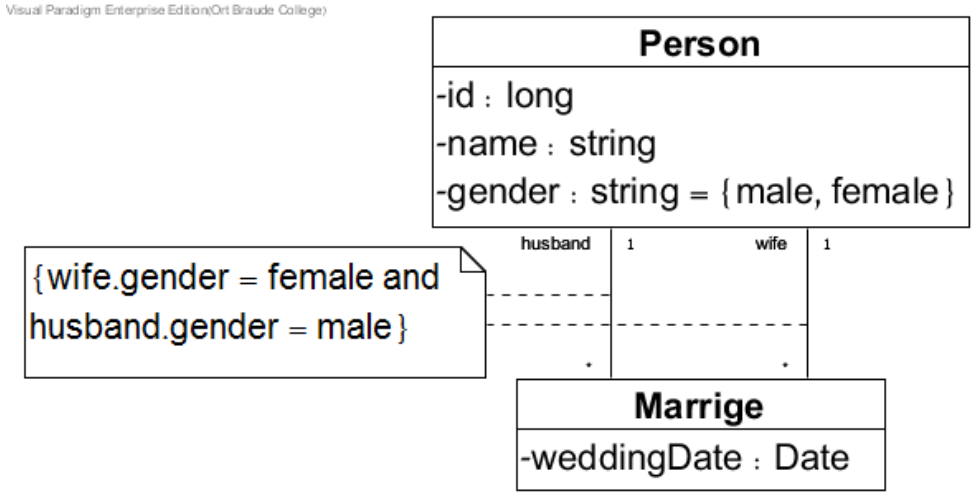
אילוצים

- יש מספר סוגי אילוצים ב- class diagram:
  - אילוץ על תכונה
  - אילוץ על קשר
  - אילוץ על מחלקה
- אילוץ יסומן ב- { }
- אילוץ על תכונה זהו מצב שבו מגדירים אילו ערכים יכולים להיות.
  - למשל:



אילוצים

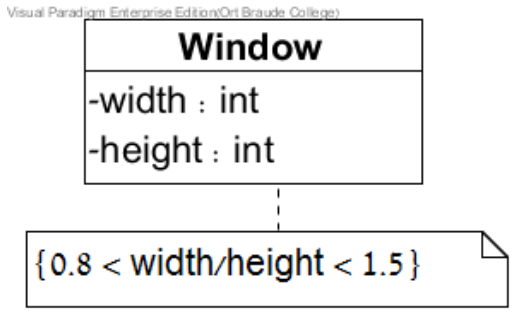
- אילוץ על קשר זהו מצב שבו מאלצים אובייקט על קשר.
  - לדוגמא:



  - אילוצים אלו יומרו לפונקציות בשכבת ה- GUI (ממשק משתמש)

אילוצים

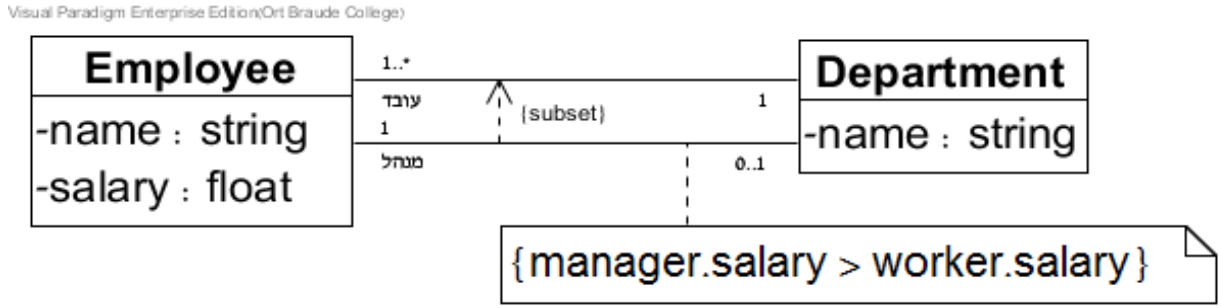
- אילוץ על מחלקה זהו מצב שבו מאלצים את האובייקט.
  - למשל:



  - אילוצים אלו יומרו לפונקציות בשכבת ה- GUI (ממשק משתמש)

אילוצים מובנים

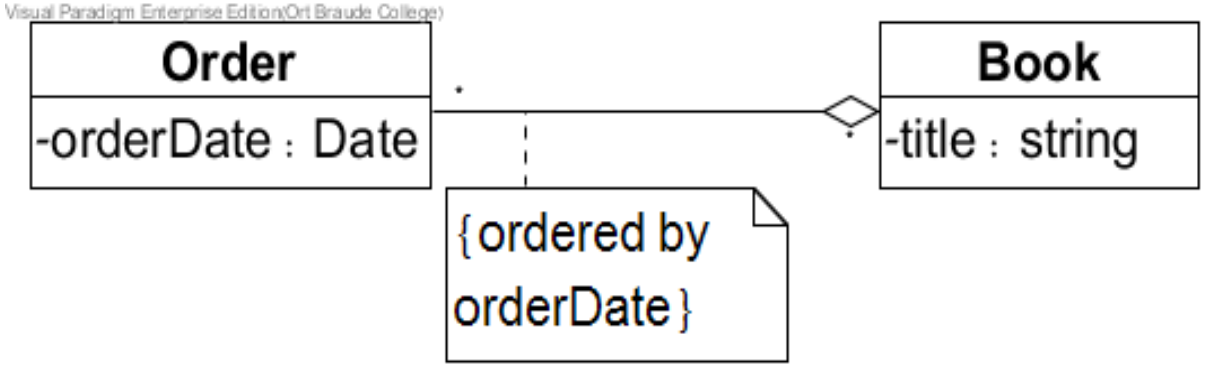
- סט של אילוצים מוגדרים.
- אילוץ subset
  - כאשר יש יותר מקשר אחד בין אובייקטים יכול להיות שאובייקט בקשר אחד הוא תת-רשימה של אובייקט בקשר השני.
  - לדוגמא: מנהל הוא חלק מעובדי המחלקה



  - אילוצים אלו יומרו לפונקציות בשכבת ה- GUI (ממשק משתמש)

אילוצים מובנים

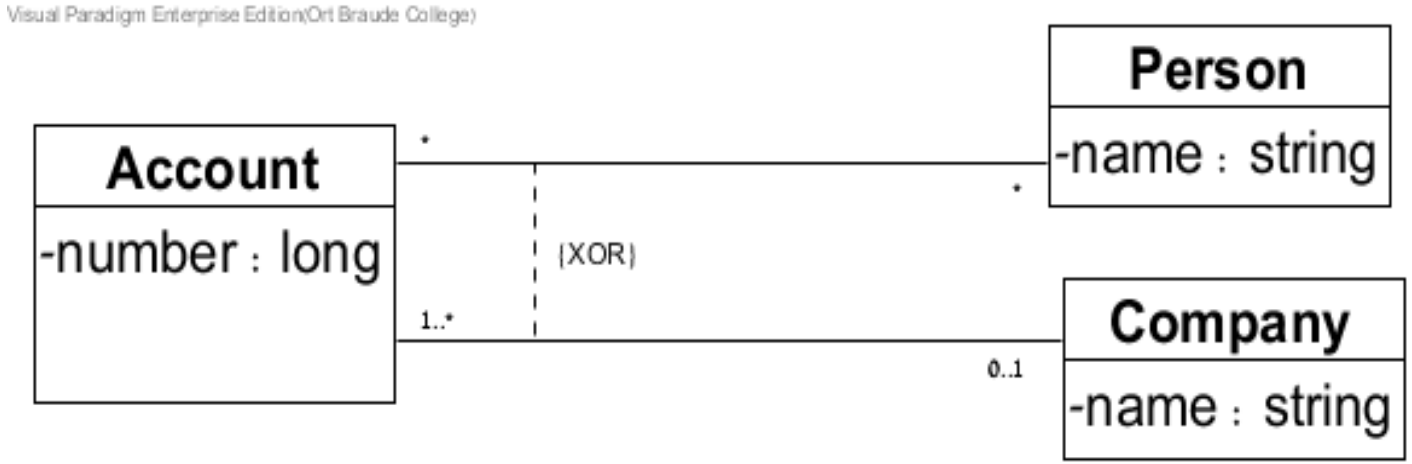
- אילוץ ordered
  - כאשר יש משמעות לסדר חיבור האובייקטים.
  - לדוגמא: תור הזמנות.



  - במחלקה Book תהיה רשימה ממוינת של אובייקטים ממחלקה Order לפי orderDate

אילוצים מובנים

- אילוץ XOR/OR על קשרים
  - כאשר יש למחלקה קשר לשני מחלקות שונות והיחס ביניהן הוא XOR/OR.
  - למשל: חשבון בנק יכול להיות שייך ללקוח פרטי או לחברה



  - אילוצים אלו יומרו לפונקציות בשכבת ה- GUI (ממשק משתמש)

תוכן ההרצאה

- מבוא
- מוטיבציה
- מרכיבי הדיאגרמה
- דוגמא