

יסודות מדעי המחשב – תרגיל בית 4 מתרגל אחראי: משה דוידיאן

הנחיות כלליות:

- מועד אחרון להגשה: כמפורסם בתיבת ההגשה ב-Moodle.
- מטרת התרגיל הינה לתרגל כתיבה, ולרכוש מיומנות בכלים שנלמדו עד כה בכיתה.
- קראו את העבודה מתחילתה ועד סופה לפני שאתם מתחילים לפתור אותה. ודאו שאתם מבינים את כל המשימות. רמת הקושי של המשימות אינה אחידה.
- את התרגיל יש לפתור לבד! לרבות איסור שימוש בכלים טכנולוגיים כגון ChatGPT.
- בתיבת ההגשה במערכת ה-VPL, ישנו קובץ שלד לכתיבת הקודם שלכם. כתבו הפתרון שלכם במקום המתאים.
- מלבד פקודת ה-pass, אין למחוק שום קטע קוד הנמצא בשלד. עליכם רק להוסיף את הפתרון שלכם בתוכו.
- אין לשנות את שם/שמות ה-Requested files.
- השאלות יבדקו באופן אוטומטי. הפלט שעליכם להחזיר בכל תרגיל צריך להיות בדיוק כפי שנדרש. כמו כן, באופן אקראי יבדקו גם עבודות באופן ידני.
- כאשר תבוצע בדיקה ידנית, תתבצע גם בדיקת Readability - שימו לב שאתם משתמשים בשמות משתנים אינפורמטיביים וכותבים הערות בכל סעיף.
- בדיקה עצמית: כדי לוודא את נכונותן ואת עמידותן של הפונקציות לקליטים שונים, בכל שאלה הריצו אותן עם מגוון קליטים: אלה שמופיעים בדוגמאות וקליטים נוספים עליהם חשבתם. וודאו כי הפלט נכון. הבדיקה תתבצע על מגוון דוגמאות ולא בהכרח אלה שיינתנו פה.
- ניתן להשתמש בחומר הנלמד עד לפרסום העבודה ורק בחומר הזה.
- אין להשתמש בחבילות או מודולים חיצוניים (דוגמאת math), למעט מקרים שבהם צוין אחרת במפורש.
- במידה שלא צוין אחרת, יש להניח את נכונות הקלט על פי תיאור המשימה.
- על מנת לקבל ניקוד מלא, יש לענות נכונה על כל השאלות במסמך זה.
- במידה שלא עניתם על שאלה מסוימת, נא מלאו את הפונקציה בכל מקרה על מנת שהקוד שלכם יצליח לעבוד.

הנחיות חשובות למשימה זאת:

- במימוש חלק מהמחלקות יהיה עליכם לממש מתודות מעבר לאלו שמפורטות במסמך זה.
 - ניתן להשתמש בספריות random ו-datetime בלבד.
 - במתודות בהם יש להחזיר מחרוזת מומלץ להשתמש ב"העתק-הדבק" מתוך מסמך זה (יכול לחסוך טעויות של אותיות קטנות/גדולות, רווחים וכו')
 - חלק איסור מוחלט להשתמש במשתנים גלובליים.
 - חל איסור מוחלט להגדיר פונקציות בתוך פונקציות.
 - חל איסור מוחלט על שימוש בפונקציות built-in בשפה כגון sum, max וכד'. ניתן לשאול בפורום אם ישנה אי בהירות לגביי פונקציה מסוימת.
 - באופן כללי, אנו לא בודקים את יעילות הקוד שלכם, אך לא יינתן ציון על בדיקה שחרגה את זמן הריצה המקסימלי של מערכת הבדיקה או עומק הרקורסיה המקסימלי.
 - אנו מרשים לעצמנו לבדוק רק חלק מהנחיות האלה בבדיקה הראשונית, ובבדיקה הסופית לבדוק את כל ההנחיות. לדוגמה, ייתכן שבדיקה הראשונית לא ייבדק שימוש במשתנים גלובליים, אך בבדיקה הסופית כן.
- אנא עקבו אחר ההנחיות על מנת להימנע מאי נעימויות בהמשך. כזכור, אין ערעורים פרטניים.**

ניהול בית גידול לציפורים





עליכם לפתח מערכת לניהול בית גידול לציפורים.

בבית הגידול ישנם כלובים שונים וכל כלוב מכיל ציפורים מסוגים שונים.

לכל אחד מהאובייקטים – בית גידול, כלוב וציפור יש מגוון מאפיינים כפי שיפורט בהמשך.

בבית הגידול ניתן לגדל מספר זני ציפורים:

טבלה 1 זני הציפורים בבית הגידול

תוכים		פינקים	
ציפור אהבה (L)	תוכן (B)	גולדיאן פינק (G)	זברה פינק (Z)
			

הערות:

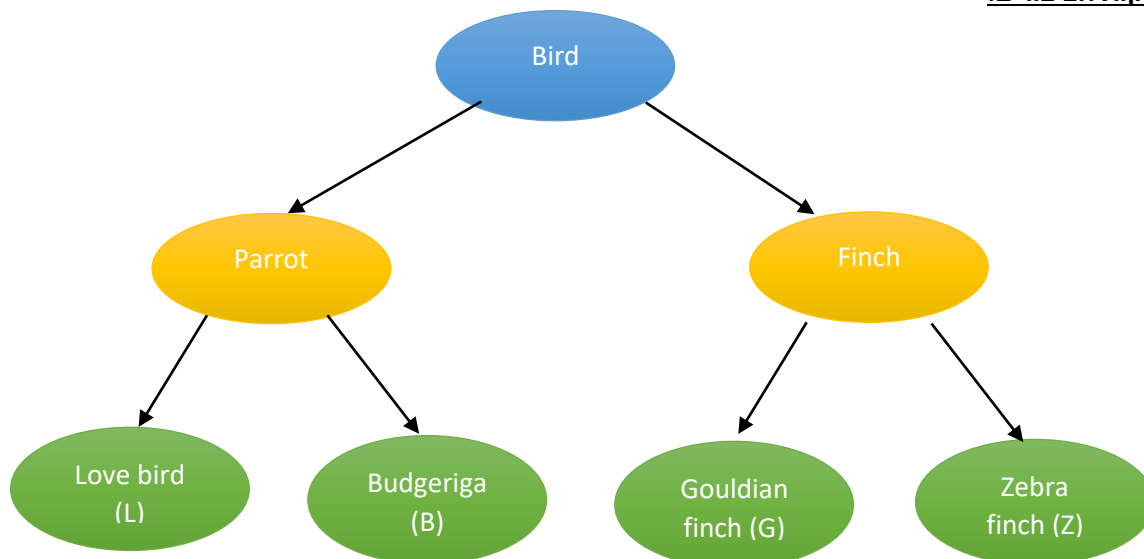
לא ניתן להניח חוקיות קלט – יש לבדוק את הנתונים הנשלחים לבנאים, במידה ואחד הנתונים לא תקין יש לזרוק שגיאה מסוג ValueError עם הודעה לבחירתכם.

ניתן להוסיף מאפיינים נוספים לכל מחלקה לבחירתכם.

אין צורך לבדוק שקוד כלוב או קוד ציפור הוא יחודי.

ניתן להניח שלא מנסים להוסיף את אותה ציפור לשני כלובים שונים ולא מנסים להוסיף את אותו הכלוב פעמיים.

מבנה מחלקות הציפורים:



פירוט המחלקות:

הנחיות לבניית המחלקה ציפור:

גיל הציפור מחושב כהפרש בין התאריך הנוכחי לתאריך הבקיעה. עליכם לחשב את מספר הימים בין התאריכים (ניתן להיעזר בפונקציות של ספריית datetime) ולאחר מכן להמיר את המספר שהתקבל לשנים, חודשים וימים. ניתן להניח בכל שנה 365 ימים ובכל חודש 30 ימים.

קוד ציפור (ID) יכול להכיל רק ספרות ואותיות גדולות.

שם המחלקה: Bird

שם המתודה	תיאור
<code>__init__(self, ID: str, btype: str, birth: str, colors: list, foodtype: str, volume: int):</code>	מאתחלת את נתוני הציפור: קוד ציפור - ייחודי (ID) סוג הציפור - אחד מהסוגים מטבלה 1. תאריך בקיעת הציפור – מחרוזת בפורמט DD:MM: YYYY צבעי הציפור – רשימה של צבעים המופיעים בציפור. לציפור יש לפחות צבע אחד. סוג מזון - "Seeds" או "Corn". נפח כלוב דרוש לציפור.
<code>get_age(self)</code>	מחזירה את גיל הציפור - שנים חודשים וימים לפי הדוגמה: ציפור בת 3 שנים, 5 חודשים ו 24 ימים: "3Y, 5M, 24D"
<code>get_type(self)</code>	מחזירה את סוג הציפור.
<code>get_colors(self)</code>	מחזירה את רשימה הצבעים של הציפור.
<code>get_food_type(self)</code>	מחזירה את סוג המזון של הציפור.
<code>get_volume(self)</code>	מחזירה את נפח הכלוב הדרוש לציפור.
<code>__str__(self)</code>	מדפיסה את פרטי הציפור. דוגמה בנספח.
אופרטורים (לא נמצאים בשלד – אתם צריכים להוסיף)	
<code>==</code>	אופרטור השוואה, מחזיר True אם שני הציפורים מאותו הסוג (False אחרת) במידה והאובייקט השני (other) אינו מסוג ציפור יש לזרוק שגיאה לבחירתכם.
<code>></code>	אופרטור "גדול ממש", מחזיר True אם מספר הצבעים שנמצאים בציפור שמפעילה את האופרטור גדול ממש ממספר הצבעים בציפור השנייה (False אחרת). במידה והאובייקט השני (other) אינו מסוג ציפור יש לזרוק שגיאה לבחירתכם.
<code><</code>	אופרטור "קטן ממש", מחזיר True אם מספר הצבעים שנמצאים בציפור שמפעילה את האופרטור קטן ממש ממספר הצבעים בציפור השנייה (False אחרת). במידה והאובייקט השני (other) אינו מסוג ציפור יש לזרוק שגיאה לבחירתכם.
ניתן / צריך להוסיף מתודות עזר	

הערות:

ניתן להניח שצבעים חוקיים יתקבלו עם אות גדולה בהתחלה וכל שאר התווים יהיו אותיות קטנות.

ניתן להיעזר בפונקציות של ספריית datetime



הנחיות לבניית המחלקה פינק (Finch):

פינקים ישנים בקנים, ככל שהפינק יותר גדול (מבוגר) הוא צריך יותר מקום.

הפינק בונה את הקן שלו לפי מספר השנים שהוא חי.

ציור קן הפינק מורכב משני קווים '||' (קווים אנכיים) בניהם קווים תחתונים ' _ ' לפי מספר השנים שחי הפינק.

לדוגמה: פינק שמספר השנים שהוא חי הוא 3 (למשל 3Y, 2M, 15D) בונה קן: " _ _ _ " (ללא רווחים, הרווחים הם רק להדגשה שיש 3 קווים תחתונים)

שם המחלקה: Finch

שם המתודה	תיאור
<code>__init__(self, ID: str, btype: str, birth: str, colors: list, volume: int)</code>	מאתחלת את נתוני הציפור: קוד ציפור ייחודי (ID) סוג הציפור. תאריך בקיעת הציפור. צבעי הציפור – רשימה של צבעים המופיעים בציפור. לציפור יש לפחות צבע אחד. סוג מזון מאתחל ל "Seeds" נפח כלוב דרוש לציפור.
<code>nest_building (self)</code>	מחזירה רשימה המכילה ציור של קן כמתואר לעיל. כל איבר ברשימה הוא תו בציור.
ניתן / צריך להוסיף מתודות עזר	



הנחיות לבניית המחלקה תוכי (Parrot):

תוכים ישנים בבתי עץ, ככל שהתוכי יותר גדול (מבוגר) הוא צריך יותר מקום.

התוכי מחפש את הבית שלו לפי מספר השנים שהוא חי.

ציור הבית של התוכי הוא ריבוע המורכב מהתו כוכבית (*) לפי מספר השנים שחי התוכי.

לדוגמה: תוכי שמספר השנים שהוא חי הוא 4 (למשל 4Y, 2M, 15D) מחפש בית כזה:

(הטבלה נועדה רק כדי להדגים את המיקומים, הציור צריך לכלול רק כוכביות)

*	*	*	*
*			*
*			*
*	*	*	*

*	*
*	*

התוכי מתחיל לחפש בית רק בגיל שנתיים ולכן הבית הקטן ביותר הוא

שם המחלקה: Parrot

שם המתודה	תיאור
<code>__init__(self, ID: str, btype: str, birth: str, colors: list, volume: int):</code>	מאתחלת את נתוני הציפור: קוד ציפור ייחודי (ID) סוג הציפור. תאריך בקיעת הציפור. צבעי הציפור – רשימה של צבעים המופיעים בציפור. לציפור יש לפחות צבע אחד. סוג מזון מאתחל ל "Corn" נפח כלוב דרוש לציפור.
<code>find_nestbox (self)</code>	מחזירה רשימה דו-ממדית המכילה ציור של בית התוכי כמתואר לעיל. כל איבר ברשימה הוא תו בציור.
ניתן / צריך להוסיף מתודות עזר	



הנחיות לבניית המחלקה זברה פינק (Zebra Finch):

רשימת צבעים חוקיים לציפור – ['Brown', 'Orange', 'Black', 'White']

שם המחלקה: Zebrafinch

שם המתודה	תיאור
def __init__(self, ID: str, birth: str, colors: list)	מאתחלת את נתוני הציפור: קוד ציפור ייחודי (ID) סוג הציפור. תאריך בקיעת הציפור. צבעי הציפור – רשימה של צבעים המופיעים בציפור. לציפור יש לפחות צבע אחד. סוג מזון מאותחל ל "Seeds" נפח כלוב דרוש לציפור – מאותחל ל 27,000 סמ"ק
get_type(self)	מחזירה את סוג הציפור – "Zebra Finch".
jump(self)	מחזירה את המשפט: "I like to jump"
__str__(self)	מדפיסה את פרטי הציפור. דוגמה בנספח.
ניתן / צריך להוסיף מתודות עזר	



הנחיות לבניית המחלקה גולדיאן פינק (Gouldian finch):

רשימת צבעים חוקיים לציפור – ['Red', 'Green', 'Blue', 'Yellow', 'Orange', 'Black', 'Purple', 'White']

שם המחלקה: Gouldianfinch

שם המתודה	תיאור
def __init__(self, ID: str, birth: str, colors: list)	מאתחלת את נתוני הציפור: קוד ציפור ייחודי (ID) סוג הציפור. תאריך בקיעת הציפור. צבעי הציפור – רשימה של צבעים המופיעים בציפור. לציפור יש לפחות צבע אחד. סוג מזון מאותחל ל "Seeds" נפח כלוב דרוש לציפור – מאותחל ל 96,000 סמ"ק עוצמת שירה – מספר שנקבע רנדומלית בעת יצירת האובייקט, מספר שלם בין 1 ל 10.
get_type(self)	מחזירה את סוג הציפור – "Gouldian Finch".
sing(self)	מחזירה את המשפט: "I like to sing" ואת עוצמת השירה.
__str__(self)	מדפיסה את פרטי הציפור. דוגמה בנספח.

ניתן / צריך להוסיף מתודות עזר



הנחיות לבניית המחלקה תוכן (Budgerigar):

רשימת צבעים חוקיים לציפור – ['Green', 'Blue', 'Yellow', 'Gray', 'Purple', 'White']

שם המחלקה: Budgerigar

שם המתודה	תיאור
<code>def __init__(self, ID: str, birth: str, colors: list)</code>	מאתחלת את נתוני הציפור: קוד ציפור ייחודי (ID) סוג הציפור. תאריך בקיעת הציפור. צבעי הציפור – רשימה של צבעים המופיעים בציפור. לציפור יש לפחות צבע אחד. סוג מזון מאותחל ל "Corn" נפח כלוב דרוש לציפור – מאותחל ל 96,000 סמ"ק. עוצמת ציוץ – מספר שנקבע רנדומלית בעת יצירת האובייקט, מספר שלם בין 1 ל 10.
<code>get_type(self)</code>	מחזירה את סוג הציפור – "Budgerigar".
<code>tweet (self)</code>	מחזירה את המשפט: "I like to tweet" ואת עוצמת הציוץ.
<code>__str__ (self)</code>	מדפיסה את פרטי הציפור. דוגמה בנספח.
ניתן / צריך להוסיף מתודות עזר	



הנחיות לבניית המחלקה ציפור אהבה (Love Bird):

רשימת צבעים חוקיים לציפור – ['Red', 'Green', 'Blue', 'Yellow', 'Black', 'White']

שם המחלקה: Lovebird

שם המתודה	תיאור
<code>def __init__(self, ID: str, birth: str, colors: list)</code>	מאתחלת את נתוני הציפור: קוד ציפור ייחודי (ID) סוג הציפור. תאריך בקיעת הציפור. צבעי הציפור – רשימה של צבעים המופיעים בציפור. לציפור יש לפחות צבע אחד. סוג מזון מאותחל ל "Corn" נפח כלוב דרוש לציפור – מאותחל ל 120,000 סמ"ק
<code>get_type(self)</code>	מחזירה את סוג הציפור – "Love Bird".
<code>kiss (self)</code>	מחזירה את המשפט: "I like to kiss"
<code>__str__ (self)</code>	מדפיסה את פרטי הציפור. דוגמה בנספח.
ניתן / צריך להוסיף מתודות עזר	

הנחיות לבניית המחלקה כלוב:

כל כלוב יכול להכיל רק ציפורים מסוג אחד.

טווח מידות הכלוב:

אורך (ס"מ)	עומק (ס"מ)	גובה (ס"מ)
30-180	30-60	40-180

כלוב מיוצג ע"י מערך דו-ממדי של תווים לפי סוג הציפור שהוא מכיל לפי טבלה 1. (כלוב ריק מיוצג ע"י התו #)
רשימת צבעים חוקיים לכלוב – ['Silver', 'Black', 'White']
כלוב צבוע בצבע אחד בלבד.
קוד כלוב (ID) יכול להכיל רק ספרות ואותיות גדולות.
בכל כלוב ניתן להכניס מספר מוגבל של ציפורים בהתאם לנפח שלו, המספר המקסימלי תלוי בסוג הציפור, כל ציפור דורשת נפח כלוב בגודל שונה. למשל בכלוב עם מידות 60X40X90 (סה"כ נפח 216,000 סמ"ק) ניתן להכניס רק שתי ציפורים מסוג תוכן (B) כי כל אחת דורשת 96,000 סמ"ק.

שם המחלקה: Cage

שם המתודה	תיאור
<code>__init__(self, ID: str, length: int, depth: int, height: int, color: str)</code>	מתחלת את נתוני הכלוב: קוד כלוב ייחודי (ID) אורך, עומק וגובה הכלוב נמדד בסנטימטרים (לדוגמה: 50X40X80). ניתן להניח שהמידות הם מספרים שלמים בקפיות של 10 (ספרת היחידות 0, למשל לא אפשרי 47). צבע הכלוב. רשימת הציפורים בכלוב – אתחול רשימה ריקה. אתחול מערך דו-ממדי המייצג את הכלוב. כל תא במערך מייצג ריבוע של 10 ס"מ x 10 ס"מ.
<code>add_bird(self, bird)</code>	מקבלת אובייקט מסוג Bird ומוסיפה אותה לכלוב. המתודה תחזיר ערך בוליאני True אם הציפור נוספה בהצלחה ו False אחרת.
<code>get_birds(self)</code>	מחזירה רשימה שמכילה את כל הציפורים בכלוב.
<code>get_num_of_birds(self)</code>	מחזירה את מספר הציפורים בכלוב.
<code>get_cage(self)</code>	מחזירה רשימה דו ממדית המייצגת תמונה של הכלוב כפי שפורט.
<code>__str__(self)</code>	מדפיסה את קוד הכלוב, מידות הכלוב, צבע הכלוב ומספר הציפורים בכלוב. ראו דוגמאות.
ניתן / צריך להוסיף מתודות עזר	

הערות:

ניתן להניח שצבעים חוקיים יתקבלו עם אות גדולה בהתחלה וכל שאר התווים יהיו אותיות קטנות.

דוגמאות:

```
c3 = Cage('1C', 100, 40, 60, 'Black')
print(c3)
print(c3.get_cage())
```

```
Cage ID: 1C
Size: (100, 40, 60)
Color: Black
Num of birds: 0
Birds type:Empty
['#####', '#####', '#####', '#####', '#####', '#####']
```

קוד להוספת תוכן (B) לכלוב והדפסת הכלוב:

```
c3 = Cage('1C', 100, 40, 60, 'Black')
b5 = Budgerigar('B1', '11:01:2020', ['Blue', 'White'])
c3.add_bird(b5)
print(c3)
for i in c3.get_cage():
    print(i)
```

```
Cage ID: 1C
Size: (100, 40, 60)
Color: Black
Num of birds: 1
Birds type: Budgerigar
BBBBBBBBBB
BBBBBBBBBB
BBBBBBBBBB
BBBBBBBBBB
BBBBBBBBBB
BBBBBBBBBB
```

יצירת כלוב עם שני גולדיאן פינק (G) והדפסת רשימת הציפורים בכלוב:

```
c2 = Cage('1B', 140, 40, 60, 'Silver')
b3 = Gouldianfinch('G1', '03:01:2022', ['Green', 'Black'])
b4 = Gouldianfinch('G2', '13:03:2021', ['Blue', 'Orange'])

c2.add_bird(b3)
c2.add_bird(b4)

for b in c2.get_birds():
    print(b)
```

```
Bird ID: G1
Bird type: Gouldian Finch
Age: 1Y,4M,7D
Colors: ['Green', 'Black']
Food type: Seeds
Singing strength: 2
Bird ID: G2
Bird type: Gouldian Finch
Age: 2Y,1M,28D
Colors: ['Blue', 'Orange']
Food type: Seeds
Singing strength: 1
```


קוד ליצירת כלוב וניסיון להכניס 3 ציפורים כאשר מבחינת נפח יש מקום רק לשתיים:

```
c3 = Cage('1C', 60, 40, 80, 'Black')

b5 = Budgerigar('B1', '11:01:2020', ['Blue', 'White'])
print(c3.add_bird(b5))
b6 = Budgerigar('B2', '11:01:2020', ['Blue', 'White'])
print(c3.add_bird(b6))
b7 = Budgerigar('B3', '11:01:2020', ['Blue', 'White'])
print(c3.add_bird(b7))
print(c3)
```

```
True
True
False
Cage ID: 1C
Size: (60, 40, 80)
Color: Black
Num of birds: 2
Birds type: Budgerigar
```



הנחיות לבניית מחלקת בית גידול:

טווח מידות בית הגידול:

אורך (מ')	רוחב (מ')	גובה (מ')
2-10	2-6	2-3

בבית הגידול כל הכלובים תלויים על קיר אחד.

מידות הקיר הם אורך בית הגידול על גובה בית הגידול.

* לא יכולה להיות חפיפה בין מיקומי הכלובים.

בית הגידול מיוצג ע"י רשימה דו-ממדית המייצגת את קיר בית הגידול עליו תלויים הכלובים. הקיר בגודל אורך בית הגידול על גובה בית הגידול. כל איבר ברשימה מייצג ריבוע של 10 ס"מ X 10 ס"מ. בנוסף לקיר יש מסגרת המייצגת את התקרה, הרצפה ושני הקירות הצדדיים. המקום שתופס ציור המסגרת לא נכלל בגודל הקיר (כלומר, הוא בנוסף לגודל שהבנאי קיבל). תמונות לדוגמה בנספחים.

שם המחלקה: Birdroom

שם המתודה	תיאור
<code>__init__(self, length: float, width: float, height: float)</code>	מאתחלת את נתוני בית הגידול: אורך, רוחב וגובה בית הגידול נמדד במטרים (לדוגמה: 2.4x3x5). ניתן להניח שהמידות הם בדיוק של ספרה אחת אחרי הנקודה (לא אפשרי 2.56). רשימת הכלובים בבית הגידול – אתחול רשימה ריקה. אתחול הרשימה הדו-ממדית שמייצגת את קיר בית הגידול עליו תלויים הכלובים.
<code>get_cages(self)</code>	מחזירה רשימה שמכילה את כל הכלובים בבית הגידול.
<code>get_birds(self)</code>	מחזירה רשימה שמכילה את כל הציפורים בבית הגידול.
<code>get_strength(self)</code>	מחזירה מספר (אחד) המייצג את סכום עוצמת הציצים והשירה של כל הציפורים בבית הגידול (אלו ששרים או מצייצים)
<code>add_cage(self, cage, x: float, y: float)</code>	מקבלת אובייקט מסוג Cage ומיקום המיוצג ע"י x ו y היכן לתלות את הכלוב ומוסיפה את הכלוב לבית הגידול במקום המתאים במידה והמקום פנוי. ניתן להניח שמוסיפים כלוב רק לאחר שיש בו לפחות ציפור אחת. המתודה תחזיר ערך בוליאני True אם הכלוב נוסף בהצלחה ו False אחרת.
<code>get_birdroom(self)</code>	מחזירה רשימה דו ממדית המייצגת תמונה של קיר בית הגידול עליו תלויים הכלובים כפי שפורט לעיל.
<code>get_most_colorful(self)</code>	מחזירה את הציפור הצבעונית ביותר (בדיקה באמצעות שימוש באופרטור שכתבתם), אם יש מספר ציפורים עם אותו מספר צבעים, תחזיר את הראשונה ברשימה.
<code>__str__(self)</code>	תדפיס את מידות בית הגידול, מספר הכלובים ומספר הציפורים בבית הגידול. דוגמה בנספחים.
ניתן / צריך להוסיף מתודות עזר	

הערות:

x, y ביחידות מידה של מטרים. ניתן להניח שהמידות הם בדיוק של ספרה אחת אחרי הנקודה (לא אפשרי 2.56). שימו לב שלא פספסתם שורה / עמודה, למשל אם $y=0.5$ כלומר 50 ס"מ, אז הכלוב צריך להיות במרחק 5 שורות משורת התקרה (כשמציירים את קיר בית הגידול ראו נספח)

נספח

גודלו הפנימי (לא כולל המסגרת) של המערך הדו-ממדי נקבע לפי גודל בית הגידול.

עבור הקוד:

```
B = Birdroom(4, 3, 2.4)
for i in B.get_birdroom():
    for j in i:
        print(j, end=' ')
    print()
```

הפלט:



הוספת שתי ציפורים מסוג זברה פינק:

```
B = Birdroom(4, 3, 2.4)

c1 = Cage('1A', 80, 40, 60, 'Silver')

b1 = Zebrafinch('Z1', '23:02:2023', ['White', 'Black'])
b2 = Zebrafinch('Z2', '13:03:2023', ['White', 'Orange'])

c1.add_bird(b1)
c1.add_bird(b2)


for bird in c1.get_birds():
    print(bird)

B.add_cage(c1, 0.2, 0.5)

for i in B.get_birdroom():
    for j in i:
        print(j, end=' ')
    print()
```

הפלט:

```
Bird ID: Z1
Bird type: Zebra Finch
Age: 0Y,2M,20D
Colors: ['White', 'Black']
Food type:Seeds
Bird ID: Z2
Bird type: Zebra Finch
Age: 0Y,2M,2D
Colors: ['White', 'Orange']
Food type:Seeds
```



גובה 60 ס"מ לכן יש 6 שורות

אורך 80 ס"מ לכן יש 8 תווים Z בכל שורה

```
B = Birdroom(4, 3, 2)

c1 = Cage('1A', 80, 40, 60, 'Silver')

b1 = Zebrafinch('Z1', '23:02:2023', ['White', 'Black'])
b2 = Zebrafinch('Z2', '13:03:2023', ['White', 'Orange'])

c1.add_bird(b1)
c1.add_bird(b2)

B.add_cage(c1, 0.1, 0.4)

c2 = Cage('1B', 140, 40, 60, 'Silver')

b3 = Gouldianfinch('G1', '03:01:2022', ['Green', 'Black'])
b4 = Gouldianfinch('G2', '13:03:2021', ['Blue', 'Orange'])

c2.add_bird(b3)
c2.add_bird(b4)

B.add_cage(c2, 1.3, 0.7)

for bird in B.get_birds():
    print(bird)

for i in B.get_birdroom():
    for j in i:
        print(j, end=' ')
    print()
```

פלט: (ציור הקיר יופיע אחרי פירוט הציפורים)

[illegible]

הקוד: (שימו לב! הכלוב השני במקום לא חוקי – גולש מגבולות הקיר ולכן לא נוסף לבית הגידול)

```
B = Birdroom(4, 3, 2)
c1 = Cage('1A', 80, 40, 60, 'Silver')
b1 = Zebrafinch('Z1', '23:02:2023', ['White', 'Black'])
b2 = Zebrafinch('Z2', '13:03:2023', ['White', 'Orange'])
c1.add_bird(b1)
c1.add_bird(b2)
B.add_cage(c1, 0.1, 0.4)

c2 = Cage('1B', 140, 40, 60, 'Silver')
b3 = Gouldianfinch('G1', '03:01:2022', ['Green', 'Black'])
b4 = Gouldianfinch('G2', '13:03:2021', ['Blue', 'Orange'])
c2.add_bird(b3)
c2.add_bird(b4)
B.add_cage(c2, 3.4, 0.8)

for bird in B.get_birds():
    print(bird)

for i in B.get_birdroom():
    for j in i:
        print(j, end=' ')
    print()
```

הפלט:

```
Bird ID: Z1  
Bird type: Zebra Finch  
Age: 0Y,2M,14D  
Colors: ['White', 'Black']  
Food type:Seeds  
Bird ID: Z2  
Bird type: Zebra Finch  
Age: 0Y,1M,26D  
Colors: ['White', 'Orange']  
Food type:Seeds
```

```
|  
|  
|  
|  
|   Z Z Z Z Z Z Z Z  
|   Z Z Z Z Z Z Z Z  
|   Z Z Z Z Z Z Z Z  
|   Z Z Z Z Z Z Z Z  
|   Z Z Z Z Z Z Z Z  
|   Z Z Z Z Z Z Z Z  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|  
|
```

```
Process finished with exit code 0
```

קוד: הדפסת פירוט הכלובים בבית הגידול וכלוב נוסף שלא נמצא בבית הגידול:

```
B = Birdroom(4, 3, 2)

c1 = Cage('1A', 80, 40, 60, 'Silver')

b1 = Zebrafinch('Z1', '23:02:2023', ['White', 'Black'])
b2 = Zebrafinch('Z2', '13:03:2023', ['White', 'Orange'])

c1.add_bird(b1)
c1.add_bird(b2)

B.add_cage(c1, 0.1, 0.4)

c2 = Cage('1B', 140, 40, 60, 'Silver')

b3 = Gouldianfinch('G1', '03:01:2022', ['Green', 'Black'])
b4 = Gouldianfinch('G2', '13:03:2021', ['Blue', 'Orange'])

c2.add_bird(b3)
c2.add_bird(b4)

B.add_cage(c2, 1.3, 0.7)

for c in B.get_cages():
    print(c)

print("\nNot in the bird room:\n")
c3 = Cage('1C', 100, 40, 60, 'Black')

print(c3)
```

הפלט:

```
Cage ID: 1A
Size: (80, 40, 60)
Color: Silver
Num of birds: 2
Birds type:Zebra Finch
Cage ID: 1B
Size: (140, 40, 60)
Color: Silver
Num of birds: 2
Birds type:Gouldian Finch

Not in the bird room:

Cage ID: 1C
Size: (100, 40, 60)
Color: Black
Num of birds: 0
Birds type:Empty
```

קוד למציאת הציפור הצבעונית ביותר בבית הגידול.

```
B = Birdroom(4, 3, 2.4)

c1 = Cage('1A', 80, 40, 60, 'Silver')

b1 = Zebrafinch('Z1', '23:02:2023', ['White', 'Black', 'Orange'])
b2 = Zebrafinch('Z2', '13:03:2023', ['White', 'Orange'])

c1.add_bird(b1)
c1.add_bird(b2)

B.add_cage(c1, 0.2, 0.5)

c2 = Cage('1B', 140, 40, 60, 'Silver')

b3 = Gouldianfinch('G1', '03:01:2022', ['Green', 'Blue', 'Black'])
b4 = Gouldianfinch('G2', '13:03:2021', ['Blue', 'Orange'])

c2.add_bird(b3)
c2.add_bird(b4)

B.add_cage(c2, 14, 8)

print(B.get_most_colorful())
```

פלט:

```
Bird ID: Z1
Bird type: Zebra Finch
Age: 0Y,2M,20D
Colors: ['White', 'Black', 'Orange']
Food type:Seeds
```


יצירת ציפור מכל סוג והדפסת הציפור:

```
b1 = Zebrafinch('Z1', '23:02:2023', ['White', 'Black', 'Orange'])
b2 = Gouldianfinch('G1', '03:01:2022', ['Green', 'Blue', 'Black'])
b3 = Budgerigar('B1', '11:01:2020', ['Blue', 'White'])
b4 = Lovebird('L1', '11:11:2021', ['Blue', 'Red'])

print(b1)
print()
print(b2)
print()
print(b3)
print()
print(b4)
```

הפלט:

```
Bird ID: Z1
Bird type: Zebra Finch
Age: 0Y,2M,20D
Colors: ['White', 'Black', 'Orange']
Food type:Seeds

Bird ID: G1
Bird type: Gouldian Finch
Age: 1Y,4M,11D
Colors: ['Green', 'Blue', 'Black']
Food type:Seeds
Singing strength:10

Bird ID: B1
Bird type: Budgerigar
Age: 3Y,4M,4D
Colors: ['Blue', 'White']
Food type:Corn
Tweeting strength:8

Bird ID: L1
Bird type: Love Bird
Age: 1Y,6M,4D
Colors: ['Blue', 'Red']
Food type:Corn
```

הדפסת האובייקט באמצעות print:

```
print("The Bird-Room")
print(B)
print("End")
```

```
The Bird-Room
Size: (4, 3, 2.4)
Num of cages: 1
Num of birds: 2
End
```

```
print(b1.nest_building())
print(b5.find_nestbox())

for i in b1.nest_building():
    print(i,end=' ')
print()
for i in b5.find_nestbox():
    for j in i:
        print(j,end=' ')
    print()
```

```
[1, 1, 1]
[['*', '*', '*', '*', '*'], ['*', '*', '*', '*', '*'], ['*', '*', '*', '*', '*'], ['*', '*', '*', '*', '*'], ['*', '*', '*', '*', '*']]
1.1
*****
* *
* *
* *
* *
*****
```