

```

1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5
6 namespace _11
7
8 {
9     class Rectangle
10    {
11         // חברה המחלקת - members
12         // פרטיזם של המחלקת - data members
13         private int length=4;
14         private int width=10;
15
16         // פונקציות של המחלקת - ציבוריות
17         public int area() -> member
18         {
19             return length * width; // פונקציית חישוב שטח
20         }
21         public void print()
22         {
23             Console.Write("length=");
24             Console.Write(length);
25             Console.Write(" width=");
26             Console.Write(width);
27             Console.Write(" area=");
28             Console.WriteLine(area());
29         }
30         public void print(char a) // פונקציה נוספת - תומכת בפונקציה דלה
31             // והשוויה בין בפונקציה דלה
32         {
33             int i, j;
34             for (i = 0; i < length; i++)
35             {
36                 for (j = 0; j < width; j++)
37                     Console.Write(a);
38                 Console.WriteLine();
39             }
40         }
41     class Program
42     {
43         static void Main(string[] args)
44         {
45             // הגדרת מטר / אובייקט למחלקת מלבן
46             Rectangle rec1; // מ
47             // הקמת זכרון לאובייקט
48             rec1=new Rectangle();
49             // מיציר לאובייקט
50             // Rectangle rec1=new Rectangle();
51
52             // הדפסת מטר + מסגרת
53             Console.WriteLine("output:");
54             Console.WriteLine("-----");
55             Console.Write("the area is: ");

```

3
13
17
18
4Φ

```

56     // כוונתן נפוץ בז'יז'וות המחלקות הציגו דוחות בנתורה מהוירטואלי קודהה
57     // מילוי פונקציית גדרה (area) ו-Print
58     Console.WriteLine(rec1.area()); - MemberOverride (rec1)
59     rec1.print(); // כוונתן המונוטוות - היפותרים קווכחים איזו
60     // פונקציה היא מובצת
61     rec1.print('@'); - MemberOverride (rec1)
62     getch(); - Console.Read();
63
64 }
65 }
66 }
67

```

ב-ס"ד

בז' שגיים

הסביר	בעברית	המושג
צורות הtecנות בשפות Tecנות עדכניות : C# vb java c++	תכנות מכוון אובייקטיבים או Tecנות מונחה עצמים	oop - object oriented programming
מבנה המכיל נתונים וקוד (פונקציות)	מחלקה	class
המשוגנה הנוצר ע"י הקזאה בזיכרון מהמחלקה: פריט של המחלקה.	אובייקט / מופע	object instance
משתנה או פונקציה השיכים למחלקה	חבר	member
תcovנה ראשונה של c++ כשות oop : צורף נתונים וקוד	ריכוזיות	encapsulation
תcovנה שנייה של c++ כשות oop : מחלקה יכולה לרשת מחלקה אחרות.	הורשה	inheritance
תcovנה שלישית של c++ כשות oop : גמישות בין בסיס לירוש.	פולימורפיזם / רב צורתויות	polymorphism
פונקציות בשם זהה ופרמטרים שונים	העמסת פונקציות	overloading
מידור של המידע - נתונים ופונקציות - על ידי: משתנים פנימיים, הרשות, ו שימוש בנתונים דרך הקוד כגון ב-get-set	הסתתרת מידע	information hiding (encapsulation) (גם)
גישה גישה למשתני ופונקציות מחלקה	גישה / הרשות	access area
גישה חופשית לכל	הרשות ציבורית	public
גישה חסומה, אפשרות גישה רק למחלקה הנקבחת	הרשות פרטית	private

F:\C#\22\22\Program.cs

1

```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  namespace _22
6  {
7      class Rectangle
8      {
9
10         // פרטי נתונים
11         private int length;
12         private int width;
13
14         // פונקציית חישוב שטח
15         public int area()
16         {
17             return length * width;
18         }
19
20         public bool bigRectangle() // פונקציה המחזירה ערך בוליאני
21         {
22             if (length >= 5 && width >= 4 || length >= 4 && width >= 5)
23                 return true;
24             return false;
25         }
26
27         // Overloading
28
29         public void print()
30         {
31             Console.WriteLine("length=" + length + " width=" + width + " area=" + area);
32             //Console.WriteLine("length={0} width={1} area={2}", length, width, area());
33             // הדפסה זהה
34             // קביעה רוחב הדפסה ומספר ציון ליניארי, יישור לשיטות
35             // Console.WriteLine("length={0,-7} width={1,-7} area={2,6}", length, width, area());
36         }
37         public void print(char a)
38         {
39             int i, j;
40             for (i = 0; i < length; i++)
41             {
42                 for (j = 0; j < width; j++)
43                     Console.Write(a);
44                     Console.WriteLine();
45             }
46         }
47         public void print(bool all) // פרמטר בוליאני
48         {
49             Console.Write("\n rectangle: ");
50             print();
51             if (all) // if(all==true) להלן
52                 print('*');

```

```

53     }
54
55     // בונים את המבנה במבנה Constructors
56     // המבנה כולל כל הMembers
57     public Rectangle(int length,int width) // בונה המקביל פרמטרים לכל
58     {                                         המمبرס משתנים
59         this.length = length;                שמות הפרמטרים כמו שמות //
60         this.width = width;                 הוא this.length
61     }                                         member
62     //public Rectangle(int l, int w) // אין אפשרות לרשום אותה בונה כאשר
63     //{
64         length = l;
65         width = w;
66     }
67
68     // : this הפעלת בניי מבני בין שורת הכוורת לגוף עם
69     public Rectangle() : this(4, 10) { } // this
70
71     public Rectangle(int x) : this(x, x) { } // מתחאים למלבן שהוא ייבוט
72
73 //-----  

74 class Program
75 {
76     static void Main(string[] args)
77     {
78         // הגדרת מספר מופעים למחילה
79         Rectangle rec1,rec2,rec3,rec4,rec5;
80         // הקצת אובייקטים המפוזלים בנאים שונים
81         rec1 = new Rectangle(7,12); - שאלות
82         rec2 = new Rectangle(5);
83         rec3 = new Rectangle();
84         // זהה המשמה הבאה אינה יוצרת אובייקט חדש אלא יש 2 אפרוריות לגשת
85         // לאותו אובייקט
86         rec4 = rec2;
87         rec5 = new Rectangle(8, 4);
88
89         rec1.print(true); // הפרמטרים קובעים איזו פונקציה תבוצע
90         rec2.print();      - overloading
91         bool b = true; // משנה בוליאני
92         rec4.print(b); // rec2.print(b); זהה
93         if (rec5.bigRectangle())
94             rec3.print('o');
95         else rec5.print();
96
97         Console.Read();
98     }
99 }
100}
101

```

הערות סידור ופונקציות:

- לינוקים ופונקציות: `bigRectangle()`, `print(bool)`, `print(char)`.
- שימוש ב-`this`: בстр� 68, בстр� 69, בстр� 71.
- שימוש ב-`new`: בстр� 81, בстр� 82, בстр� 83.
- שימוש ב-`=`: בстр� 64, בстр� 65.
- שימוש ב-`if`: בстр� 93.
- שימוש ב-`else`: בстр� 94.
- שימוש ב-`Console.Read()`: בстр� 96.

```
1
2
3     class nishemet
4     {
5         private string name;
6
7         public string Name
8         {
9             get { return name; } הערך של name/הערך הנוכחי
10            set { name = value; } הערך החדש של name/הערך החדש
11        }
12
13         private int tz;
14
15         public int Tz
16         {
17             get { return tz; }
18             set { tz = value; }
19         }
20
21         private int gil;
22
23         public int Gil
24         {
25             get { return gil; }
26             set { gil = value; }
27         }
28
29         private bool shulam;
30
31         public bool Shulam
32         {
33             get { return shulam; }
34             set { shulam = value; }
35         }
36
37         private int tziyun;
38
39         public int Tziyun
40         {
41             get { return tziyun; }
42             set { tziyun = value; }
43         }
44
45         private int misAshray;
46
47         public int MisAshray
48         {
49             get { return misAshray; }
50             set { misAshray = value; }
51         }
52     }
53
54
```

```
1
2     class nirshemet
3     {
4         private string name;
5         public string Name
6         {
7             get { return name; }
8             set { name = value; }
9         }
10
11        private int tz;
12        public int Tz
13        {
14            get { return tz; }
15        }
16
17        private int gil;
18        public int Gil
19        {
20            get {
21                if( gil > 50 )
22                    return gil;
23                return 50;
24            }
25            set {
26                if ( value > 20 )
27                    gil = value;
28            }
29        }
30
31        private bool shulam;
32        public bool Shulam
33        {
34            get { return shulam; }
35            set { shulam = value; }
36        }
37
38        private int tziyun;
39        public int Tziyun
40        {
41            get {
42                if(Shulam==true)
43                    return tziyun;
44                return 0;
45            }
46            set {
47                if ( value>=60 && value <= 100 )
48                    tziyun = value;
49            }
50        }
51        private int misAshray;
52
53    }
54
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 namespace _44
6 {
7     // properties - מאפיינים
8
9     // properties בדרך כלל נרשום לכל Member משנה
10    // value get set - תמיילים שמורוחת
11    // אפשרות להתייחס בתוכנית לMember משנה עם שם ציבורי //
12    // ולמשנה מופעלת פונקציה. //
13    // נוכחות של שימוש יstor باسم המשנה אך עם שיליטה. //
14
15    הפונקציה set מאפשרת לשנות את הערך בMember משנה הפרט //
16    // הו א הערך החדש ש-set קיבל לשום משנה הפרט //
17    // אך ניתן לבדוק את תקינות הערך לפני הטעמה //
18
19    הפונקציה get מאפשרת להציג/לקבל את הערך של המשנה הפרט //
20    // אך ניתן לא לשום פונקציה זו או לא לסקן את המידיט //
21
22    הבונה מתחל לשמות הציבוריים לצורך תקינות //
23
24    class Date
25    {
26        // data members נתונים
27
28        private int day, month, year;
29
30
31        //פונקציות
32
33        public int daysInMonth()
34        {
35            if (month == 2)
36                return 28;
37            return month == 4 || month == 6 || month == 9 || month == 11 ? 30 : 31;
38        }
39
40
41        public void print()
42        {
43            Console.WriteLine(day + "/" + month + "/" + year);
44        }
45
46        // Constructors
47
48        public Date(int day, int month, int year)
49        {
50
51            this.year = year;
52            this.month = month;
53            this.day = day;
54        }
55    }
```

```
56     }
57     //-----
58     class Program
59     {
60         static void Main(string[] args)
61     {
62         Console.WriteLine("\n\nд1\n--");
63         Date d1 = new Date();
64         d1.print();
65         d1.Day = 20;//set day
66         d1.print();
67
68
69         Console.WriteLine("\n\nд2\n--");
70         Date d2 = new Date(23, 4, 2022);
71         d2.print();
72         if(d2.Month>1)//get month
73             Console.WriteLine("yes");
74
75
76         Console.WriteLine("\n\nд3\n--");
77         Date d3 = new Date(31,1);
78         Console.WriteLine(d3.Month);//get month
79         d3.Month = 12;//set month
80         d3.print();
81
82         Console.Read();
83     }
84 }
85 }
```

סיכום

במחלקה יש members : משתנים (data members) ופונקציות.
(אין פונקציה שאינה בתוך מחלקה כל שהיא, ואין משתנה גלובלי)

הfonkaciyot ziboriyot - public
המשתנים, ה- data members - private, לא מוכרים מחוץ למחלקה שלהם.
כל ה-members מכירם זה את זה ולכן ניתן לשלוח את ה- data members לפונקציות המחלקה.

פונקציה מעומסת - overloading יש לה תפקיד דומה לפונקציה אחרת והוא בשם זהה לפונקציה
האחרת, אך חייב להיות שונה בפרמטרים : בסוג או במספר.
בפעולת הפונקציה - סוג או מס' פרמטרים קבועים איזו פונקציה תבוצע.

הקצת אובייקט / מופיע למחלקה דורשת 2 פעולות, היכלותה בשרות החקצאה.
דוגמא להקצתה למחלקה Rectangle rec1=new Rectangle();
אתחול אובייקט לא ע"י new בוגן; Rectangle rec2=rec1;
לאחר הקצת אובייקט הגישה לממבר הנמצא בו היא ע"י האופרטור נקודת .
שם ממבר. שם אובייקט

פקודת הדפסה :
Console.WriteLine("...");
בעזרת האופרטור + ניתן לשדר מספר הדפסות בפקודה בודדת.
Console.WriteLine("a={0}",a);
ניתן לשלב משתנים בחרוזות ע"י סוגרים מסולסלים, דוגמה :
Console.WriteLine("a={0,-6} b={1,8}",a,b);
ניתן להוסיף עיצוב, דוגמה קביעת רוחב הדפסה ויישור לימין ע"י מספר חיבוי, לשם ע"י מספר שלילי.

משתנה בוליאני הוא מסוג bool, ערכו true או false. יכול להיות פרמטר בוליאני ופונקציה יכולה
להחזיר ערך בוליאני. רק למשתנה בוליאני אפשר לרשום (x).if()

משתנה מחרוזת הוא מסוג string. בשונה משפט C ניתן לרשום "abc"; s="abc" וגם s=s2;s=s3;

פונקציה בונה - constructor תפקידה לאותחל את ה- data members.
שמה הוא שם המחלקה ואין לה ערך מוחזר, אפיו לא void.
שלא יאותחלו במבנה יהיו מאופסים או עם הערכים שקבלו בשרות החקצאה.

מבנה מלאה מקבלת פרמטרים לכל ה-data members שעלייה לאותחל.
ניתן לרשום את שמות הפרמטרים זהה לשמות המمبرס אך אז שם המمبر כולל this.

מבנה חלקית מקבלת פרמטרים לחילק מה멤ברס.
היא יכולה להפעיל את הבונה המלאה : בין שורת הכוורת לגוף יש לרשום this : ולשלוח
פרמטרים : את הפרמטרים שישם וכן קבועים.

מבנה ללא פרמטרים - היא מבנה המאותחל בדרך אחרת. כגון מקובעים.
גם היא יכולה להפעיל עם this : כמויל את אחת הבונות האחרות.

this הוא שם האובייקט הנוכחי.
שימוש בו מאפשר להפעיל במבנה, וכן להבדיל בין פרמטר לממבר כאשר יש להם אותו שם.
כאשר פונקציה מקבלת פרמטר בשם המمبر - נניח a, כאשר רשום בפונקציה פונה
לפרמטר. על מנת לפנות לממבר יש לרשום a.this.
בפעולת בונה מבונה אחרת : בין שורת הכוורת לגוף יש לרשום this : ולשלוח פרמטרים.
הפרמטרים יכולים להיות קבועים או פרמטרים שהתקבלו בשרות הכוורת.

המשתנים, ה- **data members**, פרטיים - **private**, ואינס מוכרים מחוץ למחלקה שלהם. ה- **public** הפונקציות ציבוריות - **public properties**.

יש רשות למשתנים properties המאפשרים שם ציבורו למשתנה הפרט. כאשר נכניס ערך חדש לשם הציבורי ע"י השמה - תופעל (באופן סמוני) הפונקציה set למשתנה זה המקבלת את הערך החדש בתוך value, ושם אותו במשתנה הפרט. בפונקציה set אפשר לרשום בדיקות ל-value ולהציג רק ערך תקין.

בצורה זו יש נוחות שימוש רגילים במשתנה ולמעשה מופעלת פונקציה הדואגת לערך תקין. כל פניה לשם הציבורו של המשתנה בלי השמה של ערך חדש מפעילה את הפונקציה get שתפקידה לחזור את הערך של המשתנה הפרטוי.

מסקנה : set ו- get שומרות על המשתנה. ניתן ~~לעשות~~ לא להזכיר get למשתנה שלא מעוניינים להראות, או לרשום get אך לחזיר את ערכו האמיתי רק בתנאים מסוימים וכו'.

מסקנה: set ו-get שומרות על המשטנה. עיתן ~~טכני~~ לא להכין get למשטנה שלא מעוניינים להראות, או לרשום get אך להזכיר את ערכו האמתי רק בתנאים מסוימים וכו'.

כיון שהבונה גם היא מכניסה ערך למمبر משטנה, למרות שהוא יכול לגשת למשטנה הפרטי, חשוב שתשתמש דוקה לשם הציגו של המשטנה על מנת שהאთחול יהיה תקין.

// properties - מאפיינים

```
// דרך כלל נרצום לכל Member משותה properties
// המילים שモרות - value get set
// מאפשרות להתייחס בתוכנית לMember משותה עם שם ציבורי
// ולמעשה מופעלת פונקציה.
// נוחות של שימוש מישר בשם המשתנה אך עם שליטה.
```

הfonction **set** מאפשר לשנות את הערך בメンバー משתנה הפרט //
value הוא הערך החדש - **set** קיבלת לשם במשתנה הפרט //
אר ניתן לבדוק את תקינות הערך לפני ההשמה //

הfonקציה `get` מאפשרת להדפיס/לקבל את הערך של המשתנה הפרטוי //
אר ניתן לא לרשות פונקציה זו או לסנן את המידע //

```

class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("\n\nд1\n--");
        Date d1 = new Date();
        d1.print();
        → d1.Day = 20;//set
        d1.print();

        Console.WriteLine("\n\nд2\n--");
        Date d2 = new Date(23, 4, 2022);
        d2.print();
        → if(d2.Month>1)//get
            Console.WriteLine("yes");

        Console.WriteLine("\n\nд3\n--");
        Date d3 = new Date(31,1);
        → Console.WriteLine(d3.Month);//
        → d3.Month = 12;//set
        d3.print();

        Console.Read();
    }
}

```

→ AND →
→ AND →

PROPERTY($f < -$)

C#

פונקציות מועמסות

פונקציות מועמסות הן פונקציות בעלות שם זהה ותפקיד דומה. כל אחת מקבלת מספר שונה של פרמטרים. מועמסות זה אותו שם. המזיאו את זה בגין הבונים, ולכן נתנו את זה גם לפונקציות האחרות. רק ה`NEW` מפעיל את הבונה. הוא מקצה אותו בזיכרון, שולחים בו פרמטרים. יש בונה מלאה וחלקית ולפי זה שולחים פרמטרים. גם לפונקציה `EAT` WRITE נתן לשלוח גם פרט וגם מחוזות, כי היא פונקציה מועמסת.
(כמובן יודעים שהוא פונקציה עפ"י הסוגרים העגולות...)

פונקציות בונות

תפקיד הבונה הוא לשלוח ערכים למשתנים. מטרתה לאותל את כל המمبرס משתנים. היא מופעלת כאשר כתבים `NEW` (ואז גם שולחים את הפרמטרים, לא שולחים...).

לזהות אותה כי שם הפונקציה הוא שם המחלקה גם שאין לה ערך מוחזר, אפילו לא `VOID`. כאשר מקצים בפעם הראשונה עם שם המשתנה, לא מופעל הבונה. ההבדל בין כל סוג הבונות הוא מספר המשתנים הנשלחים. מפעלים בונה מבונה אחרת ע"י השורה שאחרי `THIS`. (בין שורת הכותרת לגוף יש לרשום: `THIS` ולשלוח פרמטרים). משתמשים במילה `THIS` כאשר רוצים להפעיל בונה מבונה אחרת, וכן כאשר פונקציה מקבלת פרט ששמו כשם המבר.

מאפיינים

הمبرס הם פרטיים וכן יש הרשות רק לפונקציות בתוך המחלקה להשתמש בהם. ב`MAIN` אי אפשר להשתמש בהם (והכוונה כל מי שלא במחלקה). הפונקציות הם ציבוריות, ולכן אפשר להשתמש בהם גם ב`MAIN`.

אם רוצים להשתמש במשתנים גם מחוץ למחלקה (להדים, לשנות או לבדוק), יש צורך ב`PROPERTIES`. לא חייבים לעשות תמיד תמיד את `PROPERTIES`, בד"כ עושים.

הפונקציה `SET` מופעלת ב`MAIN` ע"י = וערר חדש. לדוגמא: `D1.DAY=20`.

צר שמו נוסף, שהוא ציבורי, ושם שומר המשתנה שנitin לגשת אליו.

הפונקציות מופעלות בצורה סמייה, לא יודעים שימושים בפונקציה שהיא `SET`.

ה`VALUE` הוא הערך החדש שנשלח שיישמר במשתנה הפרטי.

`GET` משתמש לקבל את הערך הפרטי. לדוגמא: קיבל את ת.ז. התלמיד, בחשבון בנק לא ינתן תוצאה. גיל מתוקן.

אל תקזו רק משתנה ציבורי במקום המשתנה הפרטי, כיון שגם תשנו ב`MAIN` את ערכו הוא לא יבדק.

וכו, מחייב לעשוט את כל המשתנים פרטיים. כך תקין יותר. אין מה לעשות זדי.

אם ה`main` שלוח ל`SET` ערך לא תקין, ישאר הערך הקודם. (אבל אם זה היה בונה, הוא ישלח 0. היזהרו...)

אם אתם לא רוצים שישמו את הערכים, אז פשוט אל תכתבו `SET`!!!

אם אתם לא רוצים לשימושם במשתנים שלכם, אל תכתבו `GET`!!!

אם תשלחו `SET` למשתנים שלא קיימים, זה פשוט יהיה שהיאת קומפליין!

משתומים בשם ציבורי כרגיל, מדפסים, מנחים, וכו'. אבל למעשה ההשמה מפעילה `SET` וההדפסה מפעילה `GET`. עושים את זה בשביבכם! אבל בתוך הפונקציה נבדקים הערכים. האמת היא, שהמחלקה היא השולחת. היא המחליטה החלטות, כי אחרי הכל, הפונקציות האלה הם שלה, לא?

שאלות חוזרת:

פונקציות מוגבלות - overloading

- מה המשמעות של מועמסות? זכרנו בז'ר זכרנו שמיון קוגניטיבי זכרנו /**Overloading**/
 - מה הקשר בין פונקציות מועמסות לפונקציות בונות? חזרנו גם לואיסטוס כ' יש לנו מושג גיאומטריה בהקשר אובייקט ה-new מפעיל בונה מתאימה. מה הכוונה? ס"כ הפיזיות שמשתמשים בnew הכוון
 - הפקודה (Console.WriteLine(...)) תזהה שהיא הפעלה פונקצייה? והכי שהיא מועמסת! שפה פולשנית זכרנו ופונקציה

פונקציות בונות - constructors

- מה תפקיד הבונה? *members* ג' נוערים.
 - מתי מופעלת הבונה? *new members*.
 - איך מזוהים בונה? *name* שם גיא גיאן צו.
 - כאשר רושמים את הפעולה הראשונה להקצתה אובייקט, הא
 - מה ההבדל בין בונה מלאה לחקיקת ולבונת מחדל? קווים נקיין
 - *טיפשיים* בונה מבונה, ואת מי נפעיל? זה? *this*? זה? *that*? (2)
 - מתי משתמשים עם המילה השמורה *this* בקשר לבונה?

מאפיינים - properties

מחלקה ואובייקט

- מה זה מחלוקת? זכרו הציגו יוזם (ולא נציג)
 - האם נכון לומר "כל דבר הוא מחלוקת"?
 - מה זה אובייקט? מה ההבדל בין אובייקט למחלוקת?
 - תני 2 דוגמאות למחלוקת ולכל אחת 2 דוגמאות לאובייקטיים.
 - למחלוקת שוקולד, איזה אובייקט נקצתה? (חודי, זיך...)
 - קניתה 2 שוקולדים זהים. האם מדובר ב-2 אובייקטים?
 - שלוחן המורה בכיתתך, זה אובייקט או מחלוקת? (אין ו-1 ו-2 הם מסוג oved). איך ניתן לשינוי באחד יגרום
 - למה משתמש האופרטור נקודה בקשר לאובייקטים?
 - מהו this?
 - פונקציות המחלוקת מכירות את ה-data members.

לעומת זה מערך

לדוגמא

```
int[] arr1 = new int[10];
int[] arr2;
arr2 = new int[20];

int[] arr3 = new int[] {1, 2, 3, 4, 5};
int[] arr4 = {1, 2, 3, 4, 5};  
new [ ] {1, 2, 3, 4, 5} , New IP place here
```

break up to use the variable by arrays

```
int[] vector = new int[]{1, 2, 3, 4, 5};  
foreach (int iItem in vector)  
{  
    Console.WriteLine ("Current Item is {0}", iItem);  
}  
// The function gets an array and print it.
```

```
using System;

public class CArraysExample1
{
    // The function gets an array and print it.
    public static void PrintArray(int[] arr)
    {
        foreach (int iValue in arr)
        {
            Console.Write("{0} ", iValue);
        }
        Console.WriteLine();
    }

    // The function gets an array and add 2 to each element in
    // the array
    public static void ChangeArray(int[] arr)
    {
        for (int i = 0; i < arr.GetLength(0); arr[i++] += 2);
    }

    // The function creates new array and returns it
    public static int[] ReturnArray()
    {
        int[] arr = new int[]{1, 2, 3, 4, 5};
        return arr;
    }

    // Main program
    public static int Main(string[] args)
    {
        int[] arr1 = new int[10]{10, 9, 8, 7, 6, 5, 4, 3, 2, 1};
        PrintArray(arr1);
        ChangeArray(arr1);
        PrintArray(arr1);
        int[] arr2;
        arr2 = ReturnArray();
        PrintArray(arr2);
        return 0;
    }
}
```

Bini

סבבון בקסה גומינטס

מערכים רב מדדיים

ב-#C שני סוגים של מערכים רבי ממדים - מערכים מלכiniים ומערכות jagged. מערכים מלכiniים הם מערכים בהם כל השורות באותו אורך, ואילו מערכות jagged הם מערכות בהן שורות שונות יכולות להיות באורך שונה.

מערכות מלכניים:

הגדות מערך רב מידי מתבצעת בעזרת רישום פסיקים בתחום הסוגרים המרובעים, כך שייהי מקום לרשום את כל הממדים של המערך, למשל:

```
int [,] mat;
int [,] cude;
```

הקצתת המרכיבים תיעשה בצורה הבאה:

```
mat = new int[10,5];
cude = new int[3,3,3];
```

ג'ישה לאיברים במרחב דו ממדית ח'יעשה באורה היבאה:

~~mat[2,3] = 3;~~
~~cude[1,2,1] = 4;~~

פונקציית INT [,] VECTOR = NEW INT [,] מוגדרת בהגדרה
 כפונקציה שמייצרת אובייקט מסוג STRING ST;
 כפונקציה ST = CONSOLE.READLINE();
 פונקציית INT X; מוגדרת כפונקציה שמייצרת אובייקט מסוג INT.
 פונקציית X = CONVERT_TOINT32(ST); מוגדרת כפונקציה שמייצרת אובייקט מסוג INT.
 פונקציית RANDOM RND = NEW RANDOM(); מוגדרת כפונקציה RANDOM.
 פונקציית INT X; מוגדרת כפונקציה INT.
 פונקציית X = RND.NEXT(10); מוגדרת כפונקציה RANDOM.
 פונקציית X = RND.NEXT(6, 16); מוגדרת כפונקציה RANDOM.

מערך :jagged

מערך jagged הוא למעשה מערך חד ממדוי, בו כל איבר והוא מערך בפני עצמו. ראשית נקזה את המערך, ולאחר מכן נקזה כל איבר בו בנפרד.

לוגמא:

```
using System;

public class CArraysExample2
{
    // Main program
    public static int Main(string[] args)
    {
        int [] [] arr;
        arr = new int[4] [];
        arr[0] = new int[10];
        arr[1] = new int[7];
        arr[2] = new int[2];
        arr[3] = new int[24];
        arr[3][2] = 2;
        return 0;
    }
}
```

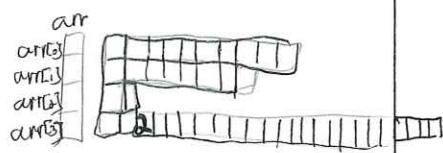
arr

arr[0]

arr[1]

arr[2]

arr[3]



length \geq arr[0] - p1 arr - p2

שיטת ומאפיינים של מערכים

למערכים יש מספר שיטות (Methods) ומאפיינים. בהם אנו יכולים להשתמש.

מִתְחָדָשׁ (Methods) מִתְחָדָשׁ

מאפיין:

תיאור: מאפיין המחויר את כמות הממדים של המעדן.

דרכן

```
int[,] cude;           ↪ תְּמִימָן  
cude = new int[7,8,9];  
Console.WriteLine(cude.Rank);      ↪ תְּמִימָן Rank ↪ תְּמִימָן  
                                ↪ תְּמִימָן Rank
```

הערך שיזדפס יהיה 3.

Length:

תיאור: מחזיר את גודל המערך, שהוא מכפלת אורכי כל הממדים שלו.

דוגמא:

```
int[,] cude;
cude = new int[10, 20, 30];
Console.WriteLine(cude.Length);
```

הערך שיודפס יהיה $10 \cdot 20 \cdot 30 = 6000$.

Clear():

תיאור: פונקציה המאפסת תחום ערכים עבור משתנים המוחזקים ערך, ומתחילה ל-null משתנים המכילים ערך התייחסות.

דוגמא:

```
int[] arr = { 1, 2, 3, 4, 5, 6, 7, 8 };
System.Array.Clear(arr, 2, 3);
for (int i = 0; i < arr.Length; ++i)
    Console.Write("{0} ", arr[i]);
Console.WriteLine();
```

יודפס: 1 2 0 0 0 6 7 8

Clone():

תיאור: פונקציה זו יוצרת העתק של המערך ומחזירה אותו.

דוגמא:

```
int[] arr1 = { 1, 2, 3, 4, 5, 6, 7, 8 };
int[] arr2 = (int[])arr1.Clone();
```

Sort():

תיאור: פונקציה זו מבצעת מיון של המערך המתkeletal כפרמטר. הפונקציה יכולה לבצע מיון של אובייקטים או מבנים בתנאי שהם ממשים את IComparable Interface (או זה נראה בהמשך המסמן).

דוגמא:

```
int[] arr = { 7, 5, 3, 4, 1, 9, 2, 8 };
System.Array.Sort(arr);
for (int i = 0; i < arr.Length; ++i)
    Console.Write("{0} ", arr[i]);
Console.WriteLine();
```

שיטה: GetLength()
תיאור: פונקציה זו מוחזירה את הגודל של מידע מסוים במערך.
דוגמא:

```
int[,] mat;
mat = new int[10,20];
Console.WriteLine("GetLength(0) = {0} and GetLength(1) = {1}",
    mat.GetLength(0), mat.GetLength(1));
```

יודפס:

GetLength(0) = 10 and GetLength(1) = 20

שיטה: IndexOf()
תיאור: הפונקציה מוחזיקה את האינדקס של המופיע הראשון של ערך מסוים. הפונקציה מקבל ערך כפרמטר, ומוחזירה את האינדקס של האיבר, או -1 אם לא נמצא איבר כזה. ניתן להפעיל פונקציה זו רק על מערכים חד ממדיים.
דוגמא:

```
int[] arr = { 4, 56, 23, 2, 43, 23, 12, 83 };
Console.WriteLine(System.Array.IndexOf(arr, 43));
```

יודפס: 6

שיטה: Binary Search()
תיאור: פונקציה זו ממחישה ערך במערך, לפי אלגוריתם של חיפוש בינארי. על המערך להיות ממוקן.
דוגמא:

```
int[] arr = { 1, 34, 45, 56, 61, 78, 88, 89, 2937 };
Console.WriteLine(System.Array.BinarySearch(arr, 61));
```

ARRAY _ REVERSE (פונקציית)

הפליה - היפוך אוריינטציית

ה-ARRAY _ REVERSE (פונקציית) מושך ל-MAIN ב-3 (1)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (2)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (3)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (4)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (5)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (6)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (7)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (8)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (9)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (10)

ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (11)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (12)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (13)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (14)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (15)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (16)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (17)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (18)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (19)
ה-ARRAY _ REVERSE מושך ל-MAIN ב-3 (20)

מערך אובייקטים

1. `new`, `משתנה ייחוס, מחסנית וערמה` (`heep`)

אובייקטים ומערכות מוקצים על ידי `new` בערמה (`heep`).
משתנה שלא מוקצה עם `new` נמצא במחסנית.

דוגמיה להקצתה בערמה:
`person p=new person(12313,"yosi");`
`p=p2;`
`new person` מעתני ייחס (זומה למצויע). `p` מצביע על האובייקט שנבנה בערמה ואוחחל במבנה.
`p` לא אוחחל על ידי `new` והוא מצביע על אותו אובייקט ש-`p` מצביע עליו. כאן נוצר רק אובייקט אחד.

2. `מערך אובייקטים חד ממדי`

הකצתה דומה להקצתה מערך חד ממדי רק במקום `int` רשום את שם המחלקה:
`person [] arr=new person[3];`

האתחול של האובייקטים במרקם `new` לכל כניסה לדוגמה:
`arr[0]=new person(3334445,"david");`
`value.print();` או `:foreach arr[i].print();`
 בלולה רשום פקודה לדוגמה:

3. `מערך אובייקטים מסוג בסיס המאותחל באובייקטים יורשים`

כלל: כשיש הורשה כגון `student` וירש מ-`person` ניתן לרשום:
`person p1=new student(), p2=new student(11223344,"dani",3);`

כלומר ניתן להקצות אובייקט יורש למשתנה ייחס מסווג בסיס.
 אך ב-`p1-p2` אפשר להפעיל רק פונקציות הנמצאות בסיס - `person` ואי אפשר להפעיל
 פונקציות הנמצאות בירש, ב-`student`.

כל זה שימושי כישיש מערך אובייקטים של הבסיס המאותחל באובייקטים מסווג הבסיס ומהיורשים.
`person [] arr=new person[10];`
`arr[0]=new student(11223344,"dani",3);`
`arr[1]=new person(765432,"moshe");`
 אה"כ נוכל לעבור בלולה על `arr` ולהפעיל פונקציות הנמצאות ב-`person`.

4. `פרמטר אסוג params`

פונקציה המקבלת מערך כפרמטר שורת הכוורת שלה היא לדוגמה:
`public static int f1(int[] arr)` :
`public static int f2(params int[] arr)` :
 אם נטיף בשורת הכוורת את המילה השמורה `params`

מה יהיה ההבדל בין `f1` ל-`f2`?

כ舍פעלים את `f1` שלוhim לה מערך.
 כ舍פעלים את `f2` שלוhim לה משתנים בודדים: אחד או 2 או 3 או יותר, כולם מסווג `int`.
 בגין הפונקציה `f2` זה נחשב ששלחו אותם במרקם. ואפשר לזרוץ עליהם בלולה בדיקת כמו ב-`f1`.
 דוגמאות להפעלה:
`f1(arr);` או `f1(vec);`
`f2(5,7);` או `f2(2,4,x+y);` או `f2(5,x,17,b,a);`

• הפרמטר `params` יכול להיות מערך אובייקטים.

• פונקציה יכולה לקבל רק `params` אחד, ואם יש עוד פרמטרים - ה-`params` יהיה האחרון.

סוגי משתנים

```

class dugma
{
private int a;
private readonly int b;
private static int c=0;
private const int masErechMusaf =17;
public dugma(int x1,int x2){a=x1; b=x2; c++;}
public int f(){int i,s=0; for(i=1;i<10;s+=i,i++); return s;} // משתנים לוקלים
// ו-s שבפונקציה f אינם מمبرס אלא הם פנימיים-ЛОקלים, הם מוקצים בפונקציה ונעלמים בסופה.//
}

הערה: המمبرס CAN כולם int אך הם יכולים להיות: string, char, וכו'

```

רמת אובייקט או רמת המחלקה	properties - get	properties - set	מאותחל ב-	מילה שומרה	סוג המשתנה
רמת האובייקט	בד"כ יש get	בד"כ יש set	בבנייה		멤בר רגיל (בדוגמה a)
רמת האובייקט	בד"כ יש get (או גו...)	אין אפשרות לשנותו כלל.	בבנייה בלבד	readonly	רייאונלי (בדוגמה b)
רמת המחלקה	בד"כ יש get	בד"כ מנוון רק במחלקה.	בבנייה או/בבנייה	static	סטטי (בדוגמה c)
רמת המחלקה	בד"כ יש get	אין אפשרות לשנותו כלל.	ב换取אה בלבד	const	קבוע (בדוגמה מס ערך מוסף)

משתנים ברמת האובייקט

המשתנים מمبرס שערכם שונה בכל אובייקט כגון אורך המלבן, שם התלמיד, צינוי התלמיד וכו' נקראים משתנים ברמת האובייקט. אם לא מעוניינים שערך ישתנה כגון מספר זהות יתאים לרשום readonly.

משתנים ברמת המחלקה

משתנים קבוע ערכו לא קשור לאובייקט מסוים. לדוגמא: 3.14 זה הוא קבוע. זהו תחليف ל-define ב-C. א. משתנה קבוע הקשור לכל האובייקטים של המחלקה לדוגמה ניתן לשמר כמה מבנים הוקזו או ב. משתנה סטטי ערכו הקשור לכל המחלקה להא תחליף לגלובלי בשפת C. כמה מבנים מתוך אילו שהוקזו שטחים גדולים-100. משתנה סטטי הוא קבוע בשפת C. ניתן להשתמש במשתנים ברמת המחלקה גם אם לא הוקצה אובייקט.

דוגמאות להקצתה ב-main: dugma d1=new dugma();

למשתנים ברמת האובייקט נרשם: d1.B ו- d1.A; שם האובייקט לפני האופרטור נקודת. שם המחלקה לפני הנקודה. dugma.MasErechMusaf ו- dugma.C אבל ברמת המחלקה נרשם: dugma.

סיכום: נשתמש ב-readonly כאשר אין גישה למחרת

נשתמש בסטטי כאשר

נשתמש בקבוע כאשר

תרגיל - המחלקה קופה. רז' 86
משתנים: שם בעל הקופה, הסכום בקופה (בשקלים), ערך דולר 3.7 נס, משתנה שבו הסכום של כל הקופות שהוקזו. כל משתנה הוא סוג אחר בטבלה.

פונקציות: properties, print, 3 בונות, print, פונקציה המחזירה את ערך הסכום בקופה בדולרים.

רשמי גם פונקציה add עם פרמטר params int.., כל המערך יתווסף לסכום בקופה.

Program.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace אדם_תלמיד
{
    // base class
    class person
    {
        private int tz;
        public int Tz
        {
            get { return tz; }
            set { tz = value; }
        }
        private string name;
        public string Name
        {
            get { return name; }
            set { name = value; }
        }
        public person() : this(123456, "israeli") { }
        public person(int tz, string name) { this.Tz = tz; this.Name = name; }
        public void print() { Console.WriteLine("tz={0} name={1}", Tz, Name); }
        public void f() { Console.WriteLine("func f"); }
    }
}

// derived class
class student : person
{
    private int kita = 1;
    public int Kita
    {
        get { return kita; }
        set { kita = value; }
    }
    public student() { }
    public student(int tz, string name, int kita) : base(tz, name) { this.Kita = kita; }
    public void print() { base.print(); Console.WriteLine("kita=" + Kita + "\n\n"); }
}

class Program
{
    static void Main(string[] args)
    {
        student s1 = new student(555555, "moshe", 2), s2 = new student();
        s1.print();
        s2.print();
        s1.f();
    }
}

```

עמוד 1

הנץ עטן כוונת רוחן

Program.cs

```
    s1.Tz = 888;  
    s1.Kita = 7;  
    s1.print();  
  
    Console.Read();  
}  
}  
}
```

תרגיל //

//
רשמי מחלקה מלבן עם ממברס אורך, רוחב, שטח והקף והפונקציות הרגילות
הורשי אותה למחלקה מגש וויספי ממברס משתנים
אזור ומחיר למטר מרובע
וכן פונקציה המחזיר את מחיר המגש
והפונקציות הרגילות //

//
רשמי תוכנית ראשית המקצת 3 מגזרים
קלטי מלוקה שטח וממחיר למגש
בדקיஇזה מהמגרשים שהקצת מתאים לлокה //

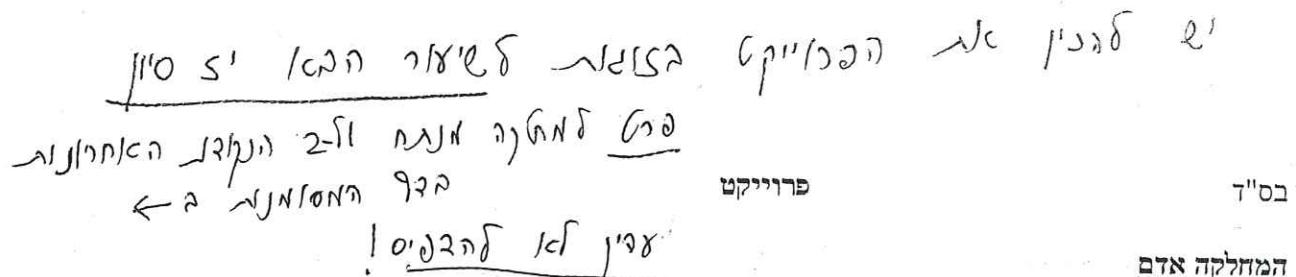
כללים //

//
היורש מכיר את המمبرס הציבוריים אך לא את הפרטאים של הבסיס
בהקצת יורש פועלת הבונה של הבסיס ואח"כ הבונה של היורש
בבונה של היורש ניתן להפעיל את הבונה של הבסיס ע"י //

כלל:

הוורשה פרטית- private- מאפשרת גישה רק למחלקה הנוכחית ולא לאף מחלקה אחרת.
הוורשה ציבורית- public- מאפשרת גישה לכל המחלקות.
הוורשה מוגנת- protected- מאפשרת גישה למחלקה הנוכחית, למחלקות היורשות אך לא למחלקות
אחרות.

הדגימי protected בתוכנית שרשמת: הויספי פונקציה מוגנת (המדפסת מלל כל שהוא) וכן משתנה פרטי
שה-properties שלו הוא protected במקום public. השתמשי בהם ביורש. בדקify שאפשר להשתמש
בם ב-main.



נתונים: שם, ת"ז (properties), גובה (לא פחות מ-0.4 מ'), משקל (הרשאה protected ל-readonly)

סכום הגבאים של כל האנשים, מסת גוף (bmi) מומלצת מינימלית - 20 ומקסימלית - 27.

פונקציות: $\text{sum}(\text{arr})$ ← $\text{getBmi}()$

3 בנות, properties, print המדפסה שם, ת"ז וגובה.

print מועמתת המקבלת פרמטר בוליאני האם להדפיס גם משקל והמלצת תזונתית.

פונקציה המחזיר את ה-bmi (למשקל 69 וגובה 1.73 bmi ה-23 הוא $69 / (1.73 * 1.73) = 23$)

פונקציה להמלצת תזונתית המדפסה: "משקל תקין" או "התחל דיאטה" או "אכול יותר".

בכל מחלוקת מ-3 המחלקות הבאות יש לרשום פונקציות: 2 בנות, properties, print

המחלקה פקיד בנק יורשת מ אדם

נתון: שם הבנק.

המחלקה רופא יורשת מ אדם

נתון: מספר מטופלים

המחלקה מנתח יורשת מרופא

נתון: האיבר שהתמחה בו (מנתח לב / מנתח מוח וכו')

ב-main

1. הכני מערך arr בגודל 8 מסוג אדם. אתחלי (לא בלולה) 2 אובייקטים מסוג אדם, 2 פקידים בנק,

2 רופאים ו-2 מנתחים. האתחול מפרמטרים.

2. קלטי שם חדש ושני את השם של הרופא הראשון וגם הוספי לגובהו 0.05 מ'.

3. הדפיסי את הגובה הממוצע של כל 8 האנשים עם מל' "הגובה הממוצע הוא:".

4. רשמי לוולה המדפסה לכל אדם במערך (כל אחד במערך הוא אדם!) את שמו והמלצת תזונתית.

5. רשמי לוולה רגילה הבודקת אם המודעות של רופאים לבריאותם גבוהה מזו של אנשים שאינם

רופאים. (ע"י השוואת bmi). הדפיסי מסקנה.

ב-program (שבו נמצא ה-main)

1. רשמי פונקציה סטית (main) המחזיר את סכום המערך.

הפעילי אותה ב-main עם 10,20,30,76 והפעלה נוספת נספתח עם 2 משתנים שמאוחלים בקלט.

2. רשמי פונקציה printArr מקבלת arr (main-arr) ומפעילה את () בולולאת foreach

3. במחלקה אדם הוסיף ב-() את המילה השמורה print():virtual

בשאר המחלקות בפונקציה הוסיף override void print():override

הפעילי את printArr ב-main לאחר ההוספה. מה קרה? מה המסקנה?

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace פרויקט_סיכון
8 {
9     class person
10    {
11        private string name;
12
13        public string Name
14        {
15            get { return name; }
16            set { name = value; }
17        }
18
19        private readonly string id;
20
21        public string Id
22        {
23            get { return id; }
24
25        }
26
27        private double height;
28
29        public double Height
30        {
31            get { return height; }
32            set { if (value >= 0.4) height = value; }
33
34        }
35        private double wight;
36
37        protected double Wight
38        {
39            get { return wight; }
40            set { wight = value; }
41
42        }
43
44        public static double sum = 0;
45
46
47        private const int bmimin = 20;
48
49        private const int bmimax = 27;
50
51        public person(string name, string id, double height, double wight)
52        {
53            this.id = id;
54            this.name = name;
55            this.height = height;
56            this.wight = wight;
```

```

57             sum += height;
58     }
59     public person(string name, string id) : this(name, id, 0, 0) { }
60     public person() : this("no name", "00000000", 0, 0) { }
61
62     public virtual void print()
63     {
64         Console.WriteLine(name + ": " + id + ": " + height);
65     }
66     public void print(bool f)
67     {
68         if (f)
69         {
70             Console.WriteLine(wight);
71             if (bmi() < bmimin)
72                 Console.WriteLine("you need eat more");
73             if (bmi() > bmimax)
74                 Console.WriteLine("you too fat");
75             else
76                 Console.WriteLine("your weight is o.k");
77
78         }
79
80
81     }
82     public double bmi() { return wight / height * height; }
83 }
84 class officer : person
85 {
86     private string bankname;
87
88     public string Bankname
89     {
90         get { return bankname; }
91         set { bankname = value; }
92     }
93     public officer(string bankname, string name, string id, double height, double wight) : base(name, id, height, wight)
94     {
95         this.Bankname = bankname;
96     }
97     public officer() : this("no bank", "no name", "00000000", 0, 0) { }
98     public override void print()
99     {
100        base.print(); Console.WriteLine(bankname);
101    }
102 }
103 class doctor : person
104 {
105     private int sum_patient;
106
107     public int Sum_patient
108     {
109         get { return sum_patient; }
110         set { sum_patient = value; }
111     }

```

```
112
113     public doctor(int sum_patient, string name, string id, double height, ↵
114         double wight) : base(name, id, height, wight)
115     {
116         this.sum_patient = sum_patient;
117     }
118     public doctor() : this(0, "no name", "00000000", 0, 0) { }
119     public override void print()
120     {
121         base.print(); Console.WriteLine(sum_patient);
122     }
123 }
124 class surgon : doctor
125 {
126     private string spec;
127
128     public string Spec
129     {
130         get { return spec; }
131         set { spec = value; }
132     }
133
134     public surgon(string spec, int sum_patient, string name, string id, ↵
135         double height, double wight): base(sum_patient, name, id, height, ↵
136         wight)
137     {
138         this.spec=spec;
139     }
140     public surgon() : this("no spec",0, "no name", "00000000", 0, 0) { }
141     public override void print()
142     {
143         base.print(); Console.WriteLine(Spec);
144     }
145 }
146
147 class program
148 {
149     public static int sum2(params int[] t)
150     {
151         int sum = 0;
152         for (int i = 0; i < t.Length; i++)
153         {
154             sum += t[i];
155         }
156         return sum;
157     }
158
159     public static void printarr(person[] arr)
160     {
161         foreach(person item in arr)
162         {
163             item.print();
164         }
165     }
166 }
```

```
165        }
166    }
167    static void Main(string[] args)
168    {
169        person[] arrp = new person[8];
170        arrp[0] = new person("Moshe Levi", "123456789", 1.7, 33);
171        arrp[1] = new person("shlomo Kohen", "987654321", 1.6, 80);
172        arrp[2] = new officer("discont", "Gadi Polak", "999999999", 1.7,
173                             65);
174        arrp[3] = new officer("mizrachi", "Dan Yeshurun", "987651234",
175                             1.6, 60);
176        arrp[4] = new doctor(100, "Nachshon Levinson", "206813904", 1.8,
177                             85);
178        arrp[5] = new doctor(50, "natan david", "674646565", 1.67, 75);
179        arrp[6] = new surgeon("heart", 50, "nachshon amindav", "208094177",
180                             1.73, 55);
181        arrp[7] = new surgeon("eyes", 50, "Michaela Dror", "056654635",
182                             1.73, 55);
183
184        arrp[4].Name = Console.ReadLine();
185        arrp[4].Height += 0.05;
186        Console.WriteLine("this avg:" + person.sum / 8);
187        for (int i = 0; i < arrp.Length; i++)
188        {
189            Console.WriteLine(arrp[i].Name);
190            arrp[i].print(true);
191        }
192        double dbmi = 0, obmi = 0;
193        for (int i = 0; i < 6; i++)
194        {
195            if (i < 4)
196                obmi += arrp[i].bmi();
197            else
198                dbmi += arrp[i].bmi();
199        }
200        if (dbmi / 2 < obmi / 4)
201            Console.WriteLine("yes,bmi doctor best");
202        Console.WriteLine(sum2(10, 20, 30, 76));
203        Console.WriteLine(sum2(Convert.ToInt32(Console.ReadLine()),
204                            Convert.ToInt32(Console.ReadLine())));
205        printarr(arrp);
206        Console.Read();
207    }
208 }
209
210 }
```

סיכום מושגים ב-#C

מחלקה ואובייקט

- מה זה מחלוקת?
 - האם נכון לומר מה זה אובייקט?
 - מה זה דוגמאות לתני 2 ותני 1?
 - למחילה שוקולד קניתי 2 שוקולד
 - שלחן המורה בפונקציות המחליל
 - למה משמש האלמנט *this*?
 - מה זה מחלוקת?

הורשה

- מתי משתמשים בהורשה?
מה המשמעות של הורשה?
מה ברווח מההורשה?
איך רושמים הורשה?
איך הבונה של היורש מפעילה את הבונה של הבסיס? מי פועלת קודם? ומה קורה אם לא
הופעלה הבונה של הבסיס?
היבאי דוגמה להורשה מירשת. متى מתאים לעשות זאת?
נתונות המחלקות קומיקס, ספר חשבון, ספר. איזו הורשה תבחרי? סעדי קז'ין ווינט חטמן ז'אנר נספחים
העובדה שפקיד יורש מאדם אפשררת
.....
אם ניתן להקנות מערכ ובו אובייקטים מסווגים שונים? אין
איזה פונקציות ניתן להפעיל כאשר המערך הוא מסווג הבסיס והאובייקטים יורשים?
אייה תועלת יכולה להיות למנעד כזו?
איך רשתות BASE ?

מה שורה ומה שוניה במושגים הבאים:

שפט C ו- C# 1

בשוויה: שפות חכבות שהבסיס שלהם זהה.

השני הנקרא **הבריטי**: הוא שפה פרוצדוראלית ואילו **C#** היא דואו - עם מחלקות וובייקטים. ניהול שונה. הבדל נוסף: ב-**C#** יש הוספות רבות כגון **foreach**, **using** ועוד.

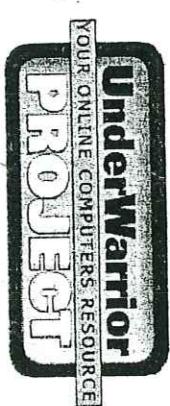
4. משתנה ייחוס שהוקצה לו אובייקט ע"י new ומשתנה ייחוס שקיבל השמה מ משתנה ייחוס אחר

18. סדרת מילים גורמת לטעות בפירושם. ניכר בכך שמיינטן וויליאם ג'ונס מילויים בפירושם של מילים יונאיות לא מילויים בפירושם של מילים ערביות. מילויים אלה מושגут על ידי מילויים יונאיים, ומיינטן וויליאם ג'ונס מילויים ערביים על ידי מילויים יונאיים.

1. תוכן עניינים

2	תוכ' עיינ'ם	.1
5	מבוא	.2
7	הכיתת אשותנה וותחבר ב-C#	.3
	Hello World	.3.1
	החבר ב-C#	.3.2
8	משתנים	.4
9	C# משפטים	.4.1
	תווים	.4.2
13	CASTING	.4.3
14	קבילה טווח המשתנים	.4.4
15	יעריכים	.5
16	אפרטורים ב-C#	.6
17	אקלרואטים שמי רשתה	.6.1
18	אקלרואטים שמי רשתה	.6.2
19	אקלרואטים ליאג'ם	.6.3
20	אלט / פולט	.7
22	משפט גראה	.8
25	IF ELSE	.8.1
25	WHILE	.8.2
26	DO WHILE	.8.3
26	FOR	.8.4
26	FOREACH	.8.5
27	פונקציית FOREACH	.8.6
28	פונקציות	.9
	מבד	.9.1
	הכניתה פונקציאקיה	.9.2
30	פונקציות חפשיות	.9.3
33	ולוקציות	.10
34	մבד	.10.1
	לכמה אסנתן	.10.2
34	http://www.underwar.co.il	

C/C++ למדריך - C#



מסמך זה הוגדר מראהו בפורטט <http://www.underwar.co.il>, לאו-שאשו מושך את המחבר.
 מחבר המשור אומת אחותה ליל ריק, שיר אונליין, שיר גאנז, שיר עידב עירוב.
 בתוסף, לו-כוננות התוקן של הנשאים המתופיעים במסתק.ע. ס. זאת, המחבר עשרה את מת'ירוב
 האטען כ"י. לסייע את המתודז'וק והמלאי בוות.

המודרך נכתב על ידי ניר אדר ד"ר שושן (nir@underwar.co.il) וויליאם פול (psclil@gmail.com).

CASTING	.15.3	השאינה	.10.3
בצאן	.15.4	סרגזיות בונת	.10.4
OBJECT OVERRIDE		DEFAULT CONSTRUCTOR	.10.5
INTERFACE	.16	סידוקיות הווות	.10.6
NAMESPACE	.16.1	PROPERTIES	.10.7
INTERFACE	.16.2	סמכים וסמכות סטטוס	.10.8
IENUMERATOR	.16.3	הנברת אגד"ם (משהו מוחיק) יפותח	.10.9
IENUMERABLE	.16.4	מכבים והעכבר פתרונות' פונקציית	.10.10
INTERFACES		מחרחת	.11
INTERFACE	.16.5	טלטול	
IDisposable	.16.6	מזהה ותוכביר	.11.1
USHED		STRINGBUILDER	.11.2
EXCEPTIONS - נזק נזק	.17	מחרחות וטוקדים	.11.3
מבוא	.17.1	שורכים	.12
APPLICATIONEXCEPTION	.17.2	גרזרת ווגמאנות ואישות	.12.1
בניגון משל	.17.3	העבירות מעכלי ליטופאץ	.12.2
OPERATORS OVERLOADING	.18	פיזיאליה מהירה מחרצת	.12.3
פעיטה אופרטורים דו-אריים	.18.1	מיכלים כמותיים	.12.4
INDEXER	.18.2	מיכלים מלבניים	.12.4.1
פונקציות מהירה	.18.3	Jagged	.12.4.2
DELEGATES	.19	שנתונות ומכושים עם טעם	.12.5
.delegate	.19.1	מביבים וטוביים מוכחים בטעפה	.13
טשייה באלגוריתם	.19.2	מביבים	.13.1
ספוף לינול חכם	.20	ENUMERATIONS	.13.2
אם ייגול החכם לא מובהך בשנייה?	.20.1	לטמיں	.13.3
הדרות השיעם	.20.2	READONLY	.13.4
עדותת האבוקטס הגדילם	.20.3	BOXING, UNBOXING	.13.5
מקטע יירוגן	.20.4	ויאוות להדרה	.13.6
מה קוראടט איסוף גבלץ?	.20.5		
ספר טיפס	.20.6		
1. רתימתם מהר	.20.6.1	C#	.14
GC.COLLECT()	.20.6.2	חרשה בשפה	.14.1
המגע אוניברס דודילם	.20.6.3	דואלא אושונה	.14.2
המגע תפוקיות הרטות	.20.6.4	ירשה אל שוד	.14.3
סימן	.21	אוחלה מלחמת גזע	.14.4
		PROTECTED	.14.5
SEALD CLASS	.14.6		
		סויומג'ו	.15
		הנברת אגד"ם	.15.1
		הנברת אגד"ם	.15.2
		http://www.underwar.co.il	

2. מבוא

מה סביבת .Net?

סביבת .Net. היא סביבת בעודודה (framework) המאפשרת מחלקות שירוט עיקן הרשפה עברת, מיציאת את התוכניות של יוניב'ר וספקת מעילו קומת התוכנכות. מסך זה אינו מודיע למתוחדים. ספק זה מיניהם שילב בשפת C++ נבר מהו שפה? כל מנה להיכר למוגנות את C++.

כasher אוות מתוכנים בשפת C# ארכו למשעה בדים מכנית מעיל התוכנוגה שא' האבבה:

- הומכית שיכוב תשמש בתולקיות מה-CLR. מיל' CLR ישיל'
- ורק שיכוב קומפל-ל-IL. בעת הראת הקוד סביבת Net.
- קוד זה להליכט לשיטת מוג�ו.

סביבת הנעודה מדידה את מגון נויל הירין האוטומוט (Garbage Collector) את כמות ה'ריכו'ן' שמנחנים בסטופ השנות מפסים פרומראים וביטם מוספים להשווים להוציאם בפעול תחומייה.

סביבת .Net. מכליה יואר מאשר את שפת C#. בנוסח C#, סביבת VB.NET-C++ וסיפות כטן VB.NET-C++ המשמשות ארכו בסביבת הוב� עיל לוח.

- מה זה סביבת .Net?
- והציגים המכבים-ב-Net. אים מהוגמים לשפת מכונה (איסטמבר). אלא לשפת ביטים – IL. ביטם הונינה. Net Framework. מאעצית את הקומפ'יז'אה האורוגה ואת הראה של התוכנית. גשה זה דומה לביינ'ים של שפת Java. מאעצית קוד Java יカリ לאראות המטרה בשתי התשויות לא לאזר שר שפה שאינה תולימת מוקה. קוד Java יカリ רחבי של מוקב רחבי של מוקב גושם מוקב גושם, ובדב'ת הנה צעד של תבנה מירטופט ב-IL דמות.
 - מיל' CLR ב-IL דמות.CLR קישור של השם פלאג'אות עשרה ביום בסיס קייזר של CLR. Common Language Runtime כל פלאג'אות-ה-Runtime ישיל שיטות מהודאות. Net. מכליה ספירית. Net השונות מהודאות. לספריה אורת מרכיבית זו גובל מושאות מטרות השבות.
 - השיבות העשירות אמוד מברחות התולקלות המתמשכות ב-IL עם השפה מברחות אגדתית שטח ספיקות לומגהן.
 - מוגנותים 'סכים' לעבר בקהלות ייחס' בין שיטות בסביבת Net – המחרחות היטוגזיות בהם הם מושמות מארות ומוקלות על המעבר ב-IL השפט.
 - טכנולוגיה לבני אינטרנט. המשגאה הפסה הדשה AI – ASP.NET – טכנולוגיה חדשה לבני אינטרנט. המשגאה הפסה הדשה AI. אין ליבור את ראתה'ם, ואפשרות שילב של הום החק'על סביבת ה-IL השפט.

3. תכנית הראשונה ותוברי בSIOT'

הפונקציה `.WriteLine()` משמשת אותנו מעתה להציג פלט. בוט התוכנית שנה קרא לה לפרט `WriteLine()` שכתובת אי הפלט את המחרשה "Hello, World". נסמן זו כפונקציית `Console.WriteLine()` המודפסת למסך.

Namespace

.Namespaces-וּן יתאפשר לארטיטים שמשות בשפה לא-תקנית. System' לדוגמה, Console' שיכריע לא-אובייקט המודפסת במאובקוט עז. או רשותם בראש המוכנית – נטען לארא לאובייקט גם כר' .System.Console.WriteLine()

2.3. תחביר בויס'

- החבר בשפת C# דומה מאד לתהבר בשפת C/C++ או שפה של מומחיים בין מיל'ם. אעל ק' של הפקודה תהייה בדבורה.
- אתם מיל'ם שמות ושמות מושגנים חיבים להיכתב/basic לעיל ווועס ב-
- אגון הכתבה בשפה:

```
using System;
public class HelloWorld
{
    public static int Main(string[] args)
    {
        Console.WriteLine("Hello, World!");
        return 0;
    }
}
```

dagshim עליהם אונן לדוח'ם מהותכיניה:

- מזהה פסיב סוף כל פקווד מה בזוקה-פסיב (.)
- מזהה הופכת הערת לתוכנית נשעת עלי ז' הרוטן // המשמש כעהרת לשורה אונן, או על ידי הסטוקן / \ בזורה התהנה, נוללה לשלוט על מספר שורה.
- בלוק בקסט השוקף סטוג'ים מושג'לים בתחלית בזופו.
- בלוק יוציא לזרות טוקצאה, ון, ון ז' או. הגדבה המתוקבלת בשפתה C# הינה מיקם ה歲'ם ותפקידיה בתבוצע ובשורה (בזורה) ופחות בשפה C. בה נהג לשים את ה岁'ם פרטיה המהוות מערך שוקן פרטם המהוות לתוכנית.
- מושות ההפיעלה. כל פרטם ומאי בא פדר במען.
- דומה לשפת C/C++ הפונקציה Main'ה מוחירה 0 בסיסים על מונת להודע למעונית הפעעליה עלי ס'ום מאיל'ה המתכוונית.

Hello, World .1.3

הדגמא הלאסוציאיינה מונה מלהב הא תוכנית המדיפסה על המנכ'ם את האזרע. דורך המוכנית הפשטה הוה נכל לראות את המרכיבים הבסיסיים ביוטר של השפה ולסודות של.

Net Class	C#	C++
System.Byte	byte	char
System.SByte	sbyte	signed char
System.Int16	short	short
System.Int32	int	int or long
System.Int64	long	_int64
System.UInt16	ushort	unsigned short
System.UInt32	uint	unsigned int or unsigned long
System.UInt64	ulong	unsigned __int64
System.Single	float	float
System.Double	double	double
System.Object	object	Object*
System.Char	char	_wchar_t
System.String	string	String*
System.Boolean	bool	bool

משתנה הוא מאטספְר האם ברכינו, לסת או נתונים שם מיהד שארה אוטומ ויכלן לשוחר בהם מדו. סבבָת Net, בעקבותיה שפה C# מכילה מונע של משתנים (objects – אובייקטים – בינה לדמויות של מולקלקט, הנקראים אובי'קטים – objects). סוגים שונים של משתנים מוגלים לרגל מידע שונו, שפה C# מודילה משתנים שוחעים לשוחר מספרי של ממש, משתנים המייעדים לשומר וויאו (אוויו), משתנים המ מייעדים לשומר מספרם ציאויל, מ. משתנים השומרים עליים ובלאיים, ומם מעתנים מוגברים. יהו Object – אובייקט – וויאן משתנה מהחרת.

מבחן החרית, הראה על משתנים זהה לארואה בשפות נויאית כ- Type <variable list>;

סבירו את Net. מצאה חישוב מחרית המשתנים לעומת שפות שקדמו לה. בשפות C/C++ נoil המשתנים אוט מודר בבלתי ומשענה העלה אתוח לשיעיה.

בסביבת Net, לעומת כל המשתנים רם בuali גודר אוות מיך נאכ. נאורה מודבר על אעד אורה מבריתת השפה, אלום למשה זה אעד משוב בוו. העקרון תומה את שפה C# הוא בסיסון ומיועת אופרטור לאלים. המשתנים קווים מקרים רבים בתם התוכנית אלה כרא עלי מחשב אבד. בשפות בון דול בערב למושגיה אורתה בה גודל המשתנים שנה מתעוררת בעיות שעונות עקיבות שהסתemo על גודל משתנים מוסיים. על מנת לאלו מוגירה באוק יוך מה גודל המשתנה, ומיירה באנטן פשטו זה את העעל מעולם המשתנים.

4. משתנים

C# בשפות 1.4

משתנה הוא מאטספְר האם ברכינו, לסת או נתונים שם מיהד שארה אוטומ ויכלן לשוחר בהם מדו. סבבָת Net, בעקבותיה שפה C# מכילה מונע של משתנים (objects – אובייקטים – בינה לדמויות של מולקלקט, הנקראים אובי'קטים – objects). סוגים שונים של משתנים מוגלים לרגל מידע שונו, שפה C# מודילה משתנים שוחעים לשוחר מספרי של ממש, משתנים המייעדים לשומר וויאו (אוויו), משתנים המ מייעדים לשומר מספרם ציאויל, מ. משתנים השומרים עליים ובלאיים, ומם מעתנים מוגברים. יהו Object – אובייקט – וויאן משתנה מהחרת.

מבחן החרית, הראה על משתנים זהה לארואה בשפות נויאית כ- Type <variable list>;

סבירו את Net. מצאה חישוב מחרית המשתנים לעומת שפות שקדמו לה. בשפות C/C++ נoil המשתנים אוט מודר בבלתי ומשענה העלה אתוח לשיעיה.

בסביבת Net, לעומת כל המשתנים רם בuali גודר אוות מיך נאכ. נאורה מודבר על אעד אורה מבריתת השפה, אלום למשה זה אעד משוב בוו. העקרון תומה את שפה C# הוא בסיסון ומיועת אופרטור לאלים. המשתנים קווים מקרים רבים בתם התוכנית אלה כרא עלי מחשב אבד. בשפות בון דול בערב למושגיה אורתה בה גודל המשתנים שנה מתעוררת בעיות שעונות עקיבות שהסתemo על גודל משתנים מוסיים. על מנת לאלו מוגירה באוק יוך מה גודל המשתנה, ומיירה באנטן פשטו זה את העעל מעולם המשתנים.

דוגמא לוגדרת משתנים בתוכנית הדיפלט

הוגמא הראה מאגרה שישמש בסיסי ב大局ו במשתנים. אינום מודרין מספר משתנים, מוצבם לתוכם ערך במתול, ויד לאור מנו מושגים אוטומטיים אס' תחכית.

```
using System;

public class VariableExample1
{
    public static int Main(string[] args)
    {
        int x = 5, y = 10;
        double dValue = 12.34;
        Console.WriteLine("Variables: ({0}, {1}, {2})", x, y, dValue);
        return 0;
    }
}
```

פלט המונחים יראה:

Variables: (5, 10, 12.34)

cohler מופיעו על השפה, שhort #define להלעון בין שני לא לביוי מומת. ככלומר, בשפה לא ניתן לטלטוטו לעזר 0 אויל false אויל 1 כלשוו. אין שווי זה מה טפייע נקי? נקי בטע למשיל בזק הד בא.

```
if (a = 3) { a = 5;
```

Variable	Size (in bits)	Range
sbyte	8	-128 to 127
short	16	-32768 to 32767
int	32	-2 ³¹ to 2 ³¹
long	64	-2 ⁶³ to 2 ⁶³
byte	8	0 to 2 ⁸
ushort	16	0 to 2 ¹⁶
uint	32	0 to 2 ³²
ulong	64	0 to 2 ⁶⁴
float	32	±1.5 × 10 ⁻⁴⁵ to ±3.4 × 10 ³⁸
double	64	±5.0 × 10 ⁻³²⁴ to 1.7 × 10 ³⁰⁸
char	16	Unicode chars
bool	8	true or false

Casting . 3.4

נית לגורר מושג המשנה או בשי להראות מושג מסוים על ידי פונקית המתגרה (Casting) (type)expression (expression נראית בתוך סוגיה).

דוגמא:

```
using System;

public class CCastingExample
{
    public static int Main(string[] args)
    {
        int a = 12, b = 5;
        double c = a / b;
        double d = (double)a / b;
        Console.WriteLine("c = (" + c + "), d = (" + d + "), c, d");
        return 0;
    }
}
```

כאר בצענו את הפעולה `a / b` והצבנו אותה ב-`c` הפעולה בזאת על שליחת `a / b` ו-`b` כארות. בטעות צבוי הדבר 2. לעומת זאת, במרקח השיל `a / b`, ובלפוניה משאנגה, מועל אותו במקאוות הזרה לוגדרן.

נק לאזע השונה בין מושג char אשר בקרה דומה ל-`'\0'` שפה C/C++.

```
char ch = 'A';
char ch2 = '!';
ch2 = ch;
```

הערה תשבה ב-`C/C++` ניק לאזע השמה גודל יותר אלו מושגיה קלה יותר, למשל השמה `char ch = '\0';` ו-`char ch = 'A';` בשתף הולט C# כההו אוסף. עד מוקה זהה ביבשה זו - שפת C# מוחמירה יותר מהשפות השונות, שבסעודה הבתוון שראה רצאה להראות מושג char למושג char שפה שונא. במודיע עקיב השמה גודלי שנייה מושג char.

תווים . 2.4

זה רם מושגיה המתואג ל-`Unicode` שירrob חווים מושגים עם מות גלובלי אותיות או סימנים.

בוגדים לתוכים בשפה C/C++ מושג char מושגים מספרים אלך רק עכברים שם מושג ערך הראה לא תבור הירוד בשפה #C:

```
char ch = 48; // פאץ
```

```
double f = 3.23;
int i = f;
```

זה אור לא יבוח קומפקטי שיכיו בוד ששי מונת שורה חד. על מונת שורה קומפקטי יש מושג.

במה מפושט:

```
double f = 3.23;
int i = (int) f;
```

שיכון הטעות אוטומטי והסoxicת'ן, פון לישא'ג קובע בתוקף אוטומטי, ובօיסטוקה הקסוציט'ן. הסופת הרקמות א' מוגר למחדר לילויו של כל קבע בתוקף 0 מוגר למחדר להגיהו של כל קבע בתוקף אוטומטי.

5. ערכים

הטבלה הבאה מכמתת ווים מיהרים, שנכתבם על ידי הופת התום \לפיניהם.

Escape Sequence	Represents
\a	Bell (alert)
\b	Backspace
\f	Formfeed
\n	New line
\r	Carriage return
\t	Horizontal tab
\v	Vertical tab
'	Single quotation mark
"	Double quotation mark
\	Backslash
?	Literal question mark
\ooo	ASCII character in octal notation
\xhhh	ASCII character in hexadecimal notation if this escape sequence is used in a wide-character constant or a Unicode string literal.
\xhhhh	Unicode character in hexadecimal notation if this escape sequence is used in a wide-character constant or a Unicode string literal.

היטויים התייחסים בשפת C# הם יווג מה אשר סוקים, והמ מילאים מאפיינים מסוימים של מתקנות אובי'קטים. מודולרתו מסכום בהן ויתר שמשנה שארם מילויים לא מושתתת ורואה שן ופונקציות הר'שי'יטים עצם (פונקציות סטטיסטיות) דוגמא ראנשונה לה'ראה מיד, המוגנתת בגאה מוגמת קבללה של העיכים המקסימלי'ם והמינימלי'ם הניתנים לה'שנה במשתנים או א' long. long מוג'ת int. long. long את הערכם עבר שאר סוג' המשנותם.

4.4. קבילה טווח המשמשת

```
using System;
public class CVariableExample2
{
    public static int Main(string[] args)
    {
        Console.WriteLine("MaxInt = {0}, MaxLong = {1}",
                           int.MaxValue, long.MaxValue);
        Console.WriteLine("MinInt = {0}, MinLong = {1}",
                           int.MinValue, long.MinValue);
        return 0;
    }
}
```

יש לשם לב כי לא כל מחרחה יש את המאפיינים שמייצגים מינ'ו ומק'ו. מאפיינים אלה קיימים באפונ אוטומטי בשעה ר'ה שמו ומייצגים הגדלים.

```
For example, wchar_t f = L"\x4e00", or wchar_t b[] =
L"The Chinese character for one is \x4e00".
```

ניר אדר nir@underwar.co.il
תאג כעט בחרבה חילק תג האופרטורים. מ

C#-ב-אופרטורים

1.6 אופרטורים של השמה

אופרטור ההשמה = הוא כבר עלי ידי מספי דוגמאות. אופרטור זה מצב לערך משתנה את

הערך הנוכחי באד המpte של היבט. בדומה לשיטות מדיניות אחרות, שיפת C# לא מאפשרות לשנות השם של הסתמה, אז רצוך ויצור נוחות.

אופרטורים הם קיימים בשפה הגדוד על ידי בינוי השפה ובונן אוסף המולוקות.

אנו האופרטורים הנחוצים בשים:

Operator category	Operators
Arithmetic	+ - * / %
Logical (boolean and bitwise)	& ^ ~ && true false
String concatenation	+
Increment, decrement	++ --
Shift	<< >>
Relational	== != < > <= >=
Assignment	= += -= *= /= %= &= = ^= <=>=
Member access	.
Indexing	[]
Cast	0
Conditional	? :
Delegate concatenation and removal	+ -
Object creation	New
Type information	is sizeof typeof
Overflow exception control	checked unchecked
Indirection and Address	* > [] &

המשתנות של הפיקוד הינו לדוגמה: 7, הרשאה גואה בעלת משלמה לשורה הגדודה:

x = 7;

באיך דומה, האופרטור -= משמשות להזנתה את התבוניה שעבד ימי של האופרטור.

לדוגמא:

int a = 2, b = 2, c, d;
c = a++;
d = ++c;

קטע הקוד לעיל היה (מבחן לוגי גט) רקוד הבא מבחרות פועלות:

אופרטורים של השוואה בשפת C#:

משמעות	אופרטור
קטן	<
גדול	>
קטן שווה	\leq
גדיל שווה	\geq
שווה	\equiv
לא שווה	\neq

3. אופרטורים לוגיים

מספר אופרטורים לוגיים (אונים וביאר'ם):

משמעות	אופרטור
NOT (אופרטור אוור)	!
AND (וגם (בגאנ')}	&&
OR (או (באי')}	

2. אופרטורים של השוואה

לכל ביט' בשפת C# יש ערך. בגין לשיטת שנות, קיימת מנגנון ביט' בין-

לאום לבין מומנטים.

בשפת C#:

- אם ערך וביט' המתחאים לו הוא שקר (FALSE) או הערך הוא מושך (TRUE).
- עבר כל ביט' השונות מה-0, מ-0-5, מ-5-0, מ-0-5 ומשם דרכ' משנתה מוגן יותר מאשר ביט' שמיינ'ו.
- למשל, ערכיו של הביט' 5 הינם אמיתיים. אם גנדר משנתה מוגן יותר מאשר ביט' שמיינ'ו.

בשפת C#:

- false, true הם ערכים לוגיים.
- ביט' לאו, היט אחד מהאים: ערך לאו,
- תאהה של ביט' המכיל אופרטור השוואה,
- אופרטורים לוגיים הם מומפעלים ביט'ים לוגיים.

מעבר אונר הוא אופרטור שפיעל עלי' ביט' אחד. אופרטור ביגאר' הוא אופרטור שפיעל ביט'	שי' ביט'ים.
מעבר האונר הוא אונר ! (משמעו מוט NOT). מתקיימת טריליה ראמת הברה:	

x	xi
false	true
true	false

למשל הbite'i (x==3) הינו ביט' לאו שערכו true .false וא true

שפת C# מחייבת אופרטורים לצור השוואה. ניתן ליצר ביט'ים לוגיים המכילים אופרטורים אלו.

עתג גאנז של האופרטורים הבינלאומיים:

/ 740 / 640

x	y	x && y	x y
false	false	false	false
false	true	false	true
true	false	false	true
true	true	true	true

הקל פועל בשיטת הפעוניות `C#` ונישם בערטת המחרחה. `ReadLine`-`WriteLine` של `Console` מוחלקה זו אחורית לפלטת הקולט והפלט של שפת `C#`. **Console.ReadLine()** מקבל דוחסיה של נתונים על המיל, הנקראת מה איסתרה לו לעצב במדעה מושמת את הפקודת היינא אל המתן. המיל מאפשר לנו לאגדו לא-ישר הפלט, לאגדות מרווחים קבועים בין פיטם, ופעולות עמוד וטפט.

ההשופטות קשורות ליעוד של התוכנויות שאנו כתובים. במדה ואונתו נקבעות תוצאותינו איזה יסתמך בה, מושגנו לשפר במתוך מהן נובא (המשמעות של *Net*). אוסף המהירות Windows Forms

בתוכנו שוכנת מנגנון הנקרא **השפה**. לא מדובר בC#-ב-ו, אלא בC#-ב-ו. מנגנון זה מאפשר לנו לשלוח ו לקבל נתונים מהשפה. לדוגמה, אם אנחנו מודים מהשפה על תכונת `Console`, נקבל תוצאה כמו זו – `System.Console`. פירוש הדבר הוא ש כל הסביבה הולכת מאפשרות לנו לקבל תוצאות מהתוכנה שבעבר היו קיימות. לדוגמה: ניתן לקבל טקסט מהתוכנה שבעבר היה קיים מה `Console`-ה. מוגדרות שיטות רבות בC#-ב-ו. אולם מוגבלם למשוך שוטרים בלבד. מוגבלים גם שיטות `ToString` ו `Equals`. מוגבלות גם שיטה `GetHashCode`.

האפרטוג || המשם יושם או (או) (ולא) (ולא)

UnderWarrior Project

22 tiny

<http://www.underwater.co.il>

卷之三

21 tiny

<http://www.underwar.co.il>

ניר אדר
nir@underwar.co.il

האגת המספר בוויסו שווה: ניקן לנקבוש אם מוסף מא' ועודיו מס'ים, על ידי ההפוך
כל'ו'ים ואחרת מהאותיות בטבלה הבאה:

מספר	אות
פ	האגת המספר 10.
א	אות מספר בראשם מהאי.
ב	אות מסוף ממשם
מ	מagenta מספר בסיסי הקודקודיימי. אם X
ל	גאגית יהיז' קסוטן, ואם X גודל האותיות ריאי' מלחלה.

דוגמא: נציג את המספרים תציג את המספרים העשויים ב-10-1 5 בoxic הסדראכמי.

```
using System;
public class CConsoleExample2
{
    public static int Main(string[] args)
    {
        int x = 5, y = 10;
        Console.WriteLine("{0, -10}{1, -10}{2, -10}{3, -10}{4, -10}{5, -10}{6, -10}{7, -10}{8, -10}{9, -10}{10, -10}{11, -10}{12, -10}{13, -10}{14, -10}{15, -10}{16, -10}{17, -10}{18, -10}{19, -10}{20, -10}{21, -10}{22, -10}{23, -10}{24, -10}{25, -10}{26, -10}{27, -10}{28, -10}{29, -10}{30, -10}{31, -10}{32, -10}{33, -10}{34, -10}{35, -10}{36, -10}{37, -10}{38, -10}{39, -10}{40, -10}{41, -10}{42, -10}{43, -10}{44, -10}{45, -10}{46, -10}{47, -10}{48, -10}{49, -10}{50, -10}{51, -10}{52, -10}{53, -10}{54, -10}{55, -10}{56, -10}{57, -10}{58, -10}{59, -10}{60, -10}{61, -10}{62, -10}{63, -10}{64, -10}{65, -10}{66, -10}{67, -10}{68, -10}{69, -10}{70, -10}{71, -10}{72, -10}{73, -10}{74, -10}{75, -10}{76, -10}{77, -10}{78, -10}{79, -10}{80, -10}{81, -10}{82, -10}{83, -10}{84, -10}{85, -10}{86, -10}{87, -10}{88, -10}{89, -10}{90, -10}{91, -10}{92, -10}{93, -10}{94, -10}{95, -10}{96, -10}{97, -10}{98, -10}{99, -10}{100, -10}{101, -10}{102, -10}{103, -10}{104, -10}{105, -10}{106, -10}{107, -10}{108, -10}{109, -10}{110, -10}{111, -10}{112, -10}{113, -10}{114, -10}{115, -10}{116, -10}{117, -10}{118, -10}{119, -10}{120, -10}{121, -10}{122, -10}{123, -10}{124, -10}{125, -10}{126, -10}{127, -10}{128, -10}{129, -10}{130, -10}{131, -10}{132, -10}{133, -10}{134, -10}{135, -10}{136, -10}{137, -10}{138, -10}{139, -10}{140, -10}{141, -10}{142, -10}{143, -10}{144, -10}{145, -10}{146, -10}{147, -10}{148, -10}{149, -10}{150, -10}{151, -10}{152, -10}{153, -10}{154, -10}{155, -10}{156, -10}{157, -10}{158, -10}{159, -10}{160, -10}{161, -10}{162, -10}{163, -10}{164, -10}{165, -10}{166, -10}{167, -10}{168, -10}{169, -10}{170, -10}{171, -10}{172, -10}{173, -10}{174, -10}{175, -10}{176, -10}{177, -10}{178, -10}{179, -10}{180, -10}{181, -10}{182, -10}{183, -10}{184, -10}{185, -10}{186, -10}{187, -10}{188, -10}{189, -10}{190, -10}{191, -10}{192, -10}{193, -10}{194, -10}{195, -10}{196, -10}{197, -10}{198, -10}{199, -10}{200, -10}{201, -10}{202, -10}{203, -10}{204, -10}{205, -10}{206, -10}{207, -10}{208, -10}{209, -10}{210, -10}{211, -10}{212, -10}{213, -10}{214, -10}{215, -10}{216, -10}{217, -10}{218, -10}{219, -10}{220, -10}{221, -10}{222, -10}{223, -10}{224, -10}{225, -10}{226, -10}{227, -10}{228, -10}{229, -10}{230, -10}{231, -10}{232, -10}{233, -10}{234, -10}{235, -10}{236, -10}{237, -10}{238, -10}{239, -10}{240, -10}{241, -10}{242, -10}{243, -10}{244, -10}{245, -10}{246, -10}{247, -10}{248, -10}{249, -10}{250, -10}{251, -10}{252, -10}{253, -10}{254, -10}{255, -10}{256, -10}{257, -10}{258, -10}{259, -10}{260, -10}{261, -10}{262, -10}{263, -10}{264, -10}{265, -10}{266, -10}{267, -10}{268, -10}{269, -10}{270, -10}{271, -10}{272, -10}{273, -10}{274, -10}{275, -10}{276, -10}{277, -10}{278, -10}{279, -10}{280, -10}{281, -10}{282, -10}{283, -10}{284, -10}{285, -10}{286, -10}{287, -10}{288, -10}{289, -10}{290, -10}{291, -10}{292, -10}{293, -10}{294, -10}{295, -10}{296, -10}{297, -10}{298, -10}{299, -10}{300, -10}{301, -10}{302, -10}{303, -10}{304, -10}{305, -10}{306, -10}{307, -10}{308, -10}{309, -10}{310, -10}{311, -10}{312, -10}{313, -10}{314, -10}{315, -10}{316, -10}{317, -10}{318, -10}{319, -10}{320, -10}{321, -10}{322, -10}{323, -10}{324, -10}{325, -10}{326, -10}{327, -10}{328, -10}{329, -10}{330, -10}{331, -10}{332, -10}{333, -10}{334, -10}{335, -10}{336, -10}{337, -10}{338, -10}{339, -10}{340, -10}{341, -10}{342, -10}{343, -10}{344, -10}{345, -10}{346, -10}{347, -10}{348, -10}{349, -10}{350, -10}{351, -10}{352, -10}{353, -10}{354, -10}{355, -10}{356, -10}{357, -10}{358, -10}{359, -10}{360, -10}{361, -10}{362, -10}{363, -10}{364, -10}{365, -10}{366, -10}{367, -10}{368, -10}{369, -10}{370, -10}{371, -10}{372, -10}{373, -10}{374, -10}{375, -10}{376, -10}{377, -10}{378, -10}{379, -10}{380, -10}{381, -10}{382, -10}{383, -10}{384, -10}{385, -10}{386, -10}{387, -10}{388, -10}{389, -10}{390, -10}{391, -10}{392, -10}{393, -10}{394, -10}{395, -10}{396, -10}{397, -10}{398, -10}{399, -10}{400, -10}{401, -10}{402, -10}{403, -10}{404, -10}{405, -10}{406, -10}{407, -10}{408, -10}{409, -10}{410, -10}{411, -10}{412, -10}{413, -10}{414, -10}{415, -10}{416, -10}{417, -10}{418, -10}{419, -10}{420, -10}{421, -10}{422, -10}{423, -10}{424, -10}{425, -10}{426, -10}{427, -10}{428, -10}{429, -10}{430, -10}{431, -10}{432, -10}{433, -10}{434, -10}{435, -10}{436, -10}{437, -10}{438, -10}{439, -10}{440, -10}{441, -10}{442, -10}{443, -10}{444, -10}{445, -10}{446, -10}{447, -10}{448, -10}{449, -10}{450, -10}{451, -10}{452, -10}{453, -10}{454, -10}{455, -10}{456, -10}{457, -10}{458, -10}{459, -10}{460, -10}{461, -10}{462, -10}{463, -10}{464, -10}{465, -10}{466, -10}{467, -10}{468, -10}{469, -10}{470, -10}{471, -10}{472, -10}{473, -10}{474, -10}{475, -10}{476, -10}{477, -10}{478, -10}{479, -10}{480, -10}{481, -10}{482, -10}{483, -10}{484, -10}{485, -10}{486, -10}{487, -10}{488, -10}{489, -10}{490, -10}{491, -10}{492, -10}{493, -10}{494, -10}{495, -10}{496, -10}{497, -10}{498, -10}{499, -10}{500, -10}{501, -10}{502, -10}{503, -10}{504, -10}{505, -10}{506, -10}{507, -10}{508, -10}{509, -10}{510, -10}{511, -10}{512, -10}{513, -10}{514, -10}{515, -10}{516, -10}{517, -10}{518, -10}{519, -10}{520, -10}{521, -10}{522, -10}{523, -10}{524, -10}{525, -10}{526, -10}{527, -10}{528, -10}{529, -10}{530, -10}{531, -10}{532, -10}{533, -10}{534, -10}{535, -10}{536, -10}{537, -10}{538, -10}{539, -10}{540, -10}{541, -10}{542, -10}{543, -10}{544, -10}{545, -10}{546, -10}{547, -10}{548, -10}{549, -10}{550, -10}{551, -10}{552, -10}{553, -10}{554, -10}{555, -10}{556, -10}{557, -10}{558, -10}{559, -10}{560, -10}{561, -10}{562, -10}{563, -10}{564, -10}{565, -10}{566, -10}{567, -10}{568, -10}{569, -10}{570, -10}{571, -10}{572, -10}{573, -10}{574, -10}{575, -10}{576, -10}{577, -10}{578, -10}{579, -10}{580, -10}{581, -10}{582, -10}{583, -10}{584, -10}{585, -10}{586, -10}{587, -10}{588, -10}{589, -10}{590, -10}{591, -10}{592, -10}{593, -10}{594, -10}{595, -10}{596, -10}{597, -10}{598, -10}{599, -10}{600, -10}{601, -10}{602, -10}{603, -10}{604, -10}{605, -10}{606, -10}{607, -10}{608, -10}{609, -10}{610, -10}{611, -10}{612, -10}{613, -10}{614, -10}{615, -10}{616, -10}{617, -10}{618, -10}{619, -10}{620, -10}{621, -10}{622, -10}{623, -10}{624, -10}{625, -10}{626, -10}{627, -10}{628, -10}{629, -10}{630, -10}{631, -10}{632, -10}{633, -10}{634, -10}{635, -10}{636, -10}{637, -10}{638, -10}{639, -10}{640, -10}{641, -10}{642, -10}{643, -10}{644, -10}{645, -10}{646, -10}{647, -10}{648, -10}{649, -10}{650, -10}{651, -10}{652, -10}{653, -10}{654, -10}{655, -10}{656, -10}{657, -10}{658, -10}{659, -10}{660, -10}{661, -10}{662, -10}{663, -10}{664, -10}{665, -10}{666, -10}{667, -10}{668, -10}{669, -10}{670, -10}{671, -10}{672, -10}{673, -10}{674, -10}{675, -10}{676, -10}{677, -10}{678, -10}{679, -10}{680, -10}{681, -10}{682, -10}{683, -10}{684, -10}{685, -10}{686, -10}{687, -10}{688, -10}{689, -10}{690, -10}{691, -10}{692, -10}{693, -10}{694, -10}{695, -10}{696, -10}{697, -10}{698, -10}{699, -10}{700, -10}{701, -10}{702, -10}{703, -10}{704, -10}{705, -10}{706, -10}{707, -10}{708, -10}{709, -10}{710, -10}{711, -10}{712, -10}{713, -10}{714, -10}{715, -10}{716, -10}{717, -10}{718, -10}{719, -10}{720, -10}{721, -10}{722, -10}{723, -10}{724, -10}{725, -10}{726, -10}{727, -10}{728, -10}{729, -10}{730, -10}{731, -10}{732, -10}{733, -10}{734, -10}{735, -10}{736, -10}{737, -10}{738, -10}{739, -10}{740, -10}{741, -10}{742, -10}{743, -10}{744, -10}{745, -10}{746, -10}{747, -10}{748, -10}{749, -10}{750, -10}{751, -10}{752, -10}{753, -10}{754, -10}{755, -10}{756, -10}{757, -10}{758, -10}{759, -10}{760, -10}{761, -10}{762, -10}{763, -10}{764, -10}{765, -10}{766, -10}{767, -10}{768, -10}{769, -10}{770, -10}{771, -10}{772, -10}{773, -10}{774, -10}{775, -10}{776, -10}{777, -10}{778, -10}{779, -10}{780, -10}{781, -10}{782, -10}{783, -10}{784, -10}{785, -10}{786, -10}{787, -10}{788, -10}{789, -10}{790, -10}{791, -10}{792, -10}{793, -10}{794, -10}{795, -10}{796, -10}{797, -10}{798, -10}{799, -10}{800, -10}{801, -10}{802, -10}{803, -10}{804, -10}{805, -10}{806, -10}{807, -10}{808, -10}{809, -10}{810, -10}{811, -10}{812, -10}{813, -10}{814, -10}{815, -10}{816, -10}{817, -10}{818, -10}{819, -10}{820, -10}{821, -10}{822, -10}{823, -10}{824, -10}{825, -10}{826, -10}{827, -10}{828, -10}{829, -10}{830, -10}{831, -10}{832, -10}{833, -10}{834, -10}{835, -10}{836, -10}{837, -10}{838, -10}{839, -10}{840, -10}{841, -10}{842, -10}{843, -10}{844, -10}{845, -10}{846, -10}{847, -10}{848, -10}{849, -10}{850, -10}{851, -10}{852, -10}{853, -10}{854, -10}{855, -10}{856, -10}{857, -10}{858, -10}{859, -10}{860, -10}{861, -10}{862, -10}{863, -10}{864, -10}{865, -10}{866, -10}{867, -10}{868, -10}{869, -10}{870, -10}{871, -10}{872, -10}{873, -10}{874, -10}{875, -10}{876, -10}{877, -10}{878, -10}{879, -10}{880, -10}{881, -10}{882, -10}{883, -10}{884, -10}{885, -10}{886, -10}{887, -10}{888, -10}{889, -10}{890, -10}{891, -10}{892, -10}{893, -10}{894, -10}{895, -10}{896, -10}{897, -10}{898, -10}{899, -10}{900, -10}{901, -10}{902, -10}{903, -10}{904, -10}{905, -10}{906, -10}{907, -10}{908, -10}{909, -10}{910, -10}{911, -10}{912, -10}{913, -10}{914, -10}{915, -10}{916, -10}{917, -10}{918, -10}{919, -10}{920, -10}{921, -10}{922, -10}{923, -10}{924, -10}{925, -10}{926, -10}{927, -10}{928, -10}{929, -10}{930, -10}{931, -10}{932, -10}{933, -10}{934, -10}{935, -10}{936, -10}{937, -10}{938, -10}{939, -10}{940, -10}{941, -10}{942, -10}{943, -10}{944, -10}{945, -10}{946, -10}{947, -10}{948, -10}{949, -10}{950, -10}{951, -10}{952, -10}{953, -10}{954, -10}{955, -10}{956, -10}{957, -10}{958, -10}{959, -10}{960, -10}{961, -10}{962, -10}{963, -10}{964, -10}{965, -10}{966, -10}{967, -10}{968, -10}{969, -10}{970, -10}{971, -10}{972, -10}{973, -10}{974, -10}{975, -10}{976, -10}{977, -10}{978, -10}{979, -10}{980, -10}{981, -10}{982, -10}{983, -10}{984, -10}{985, -10}{986, -10}{987, -10}{988, -10}{989, -10}{990, -10}{991, -10}{992, -10}{993, -10}{994, -10}{995, -10}{996, -10}{997, -10}{998, -10}{999, -10}{1000, -10}
```

אם נשנה אותה להלהות:

Console.WriteLine("0, 10){1, 10}{2, 10}{3, 10", x, y, z);

ולבסוף מתקבל תוצאה דומה במתוך המכילה מספר שורות, וගראות יי'gor טוב ייכל' ר' ר' ש'ש'ג
גאנט.

do while .3.8

בכ"מ שפטת הדרישה בשיטת **C#** לאילו שפטת **C/C++** מהתהoctave. שפטת מובנית אוניברסלית בשפת **C#** שפואת מובהת מחבר חדש, שהציג משפט **foreach** פונקציית **yield**, ואות שפטת גוראה **return**. שפטת מושג בפונקציית **yield** מוגדרת כפונקציית **foreach** שפואת מובהת מחבר חדש, שהציג משפט **foreach** פונקציית **yield**, ואות שפטת גוראה **return**.

```
for (initialization; condition; increment)  
    statement;
```

```
do
{
    // block
} while (condition);
```

foreach .5.8

לאלה דרשה זו מאפשרת לעבר על כל ה'אבירים במעלה א' ב-*collection* דרכו:

if_eise .1.8

```
if (condition) statement;
```

white
2,8

```
using System;

public class CForEachExample
{
    public static int Main(string[] args)
    {
        int[] vector = new int[]{1, 2, 3, 4, 5};
        foreach (int item in vector)
        {
            Console.WriteLine ("Current Item is {0}", item);
        }
        return 0;
    }
}
```

UnderWarrior Project

26
Tinny

<http://www.underwar.co.il>

היום קשיד זה יטוט על המלך אול הנסיך ורמונט 5 בראג האגדה שמשבעה, ונהמעיק יותר במשמעותו, ועתה ושוב רק שזו הסתירות להגדיר מערם.

9. פורציאז

9.1. מבוא

פוקאייה הרונה מתקען קוד של C#, שמרתטרים, מבעת הערך ומחזר ערך כשלשו. בצד לפועל את הפעוקאייה ש' לאר לא לה במאטעות כתובת שמה. הפעוקאייה ימלה ללבול קלט פתרון או יוות') אשר 'שפשע' על פועלו. הפעוקאייה Max שמתבלט כרך מס' או גאגמא: מביום מס' כלשהו של ערכים מס'ים, ומחרה אורה את הדואג מבויום.

בוגר לשיפוט C/C++, בשיטת C# כי הפעוקאייה ייחסו ש' ייחסו למתקלה כישריה.

שפת C# אינה מכילה "טוקציוז גולגולת", פוקצייז התמצאות מושך כל מהלך. בקיטוטם בינם בשיטות אחריות דרישות פונקציות לפחות שתי שפות C# ממענו למתנוונת להשתמש בפוקאייה סטאטית של מתקליה כדי לחשוף את התהנתנותה הראשית. בשיטת C# אין גאהה מראשת על פוקאייה יש' לממש פוקאייה מד עד כתיבת הותחה שלה.

גדירה כלות ש夷 פוקאייה:

```
[modifiers] return-type method-name([parameter-list])
```

```
public class CForEachExample2
{
    public static int Main(string[] args)
    {
        int[] vector = new int[]{1, 2, 3, 4, 5};

        foreach (int iItem in vector)
        {
            if (iItem == 3) continue;
            Console.WriteLine("Current item is {0}", iItem);
        }
        return 0;
    }
}
```

מילים שמותר שטמיינית הרשות (וכו) מילים מאפ'ים מיהדים של רפטוקאייה (למשיל הראם פוקאייה ראה פטוקאייה (למשיל גיא). הערך אומת הפטוקאייה מחרה. שם הפוקאייה רשותה הפטוקאייה מחרה את הא מ'כברין.

break-i, continue ו' פְּקָדָוִת 6.8

חויה מוקבי: מובהר אחד עד גוף מתחילה הושם שמי הון שמי בצעות הולאה. ואיטה. אטראה.

C# המשמשים בה בתרן לולאה, אטורה לשפטת, כאשר און משמשים את איטועה הינה של הולאה, ובעבר אל איטועה הינה האטה. גם הימלה השמהה break אומתות באופן C#-ליס'ם את הולאה הנוכחית בעבור לולא. לאחר מכן לולאה. לדגמא:

```
using System;
```

```
public class CForEachExample2
{
    public static int Main(string[] args)
    {
        int[] vector = new int[]{1, 2, 3, 4, 5};

        foreach (int iItem in vector)
        {
            if (iItem == 3) continue;
            Console.WriteLine("Current item is {0}", iItem);
        }
        return 0;
    }
}
```

ודסוי על התוך כל המספרים פרט למספר 3, עליי הומינת מלהג.

לוגין:

2.9 העברת פרמטרים ל함ונקציה

על מנת להעביר משתנים בין הנעלמים שארור שם הפטוקציה רshima של משתנים בפומת הבא:

```
type1 name1, type2 name2, ... typeN nameN
```

כלומר, מוכבים שיש יטיפס אחר ר' את שם המשנה מהו? אם רצץ משונה מופיע, מפרידים בפסיק בין ובו המשנה הקומן.

בון ריצעת הפטוקציה, כל המשתנים שעשויים נוכחים בשורה זו ייו' זמינים כאלו הוגדרו בתוך הפטוקאה עצמה.

המשתנים הבסיסיים (int, double, long) ווכ'ו) שעברם למספרים מסוימים לונציגות value by value כבניה תמורה. בarshal' שנעשה סעיף reference reference וראה שהם שעברם עוצב על reference מוחלט.

עם זאת, בירית המוחלט אונגה רקקה בסילע. ניקן להסביר פוטומרים בדרכיהם ונטופות על צי' שמש בתמילים שמותרות של שפת #C:

```
using System;
public class cfunctionExample1
{
    public static int Max(int x, int y)
    {
        return (x > y) ? x : y;
    }
    public static int Main(string[] args)
    {
        int x = 45, y = 100;
        Console.WriteLine("Max between {0} and {1} is {2}", x, y, Max(x,y));
        return 0;
    }
}
```

על פ' בירית המוחלט, העכלה משבירם by value מוחך לפטוקציה "שארא לא שט".

העברת **ref** בהרבה: כאשר אנו מעבירים משתנה by reference הפטוקיה מוחך לפטוקציה, והעתקה מוחך לפטוקציה. כנראה מהך, אם נשאה את הפרטש בתול' הפטוקיאיה, ישנה גם המשנה מהול' הפטוקיאיה. העברת זו שמשמשת במקרים-כגון מדברים על מוחלקות, אלים ימי מקרים ובום ונזה' לשונו גם מושגים טיסיים שמדובר-בפומת. אחת הה�גיאות הפטוקיות בוחר לker' תוא' הפטוקיאה swap היל'קוחת שעיכים ומוחלי'פה את ערכם.

אלא מוחט שיל' הפטוקיאה swap נuber מושגים שלם בשפת C# וזה שיטוש בהנברת המשנים פיניטים: משתנים המודדים בתר הפטוקיאיה, הינם מושגים ליקויים של הפטוקיאיה. סעוך הרכהה שלם הוא בתו' הפטוקיאיה בלבד, ואיל' אוור הח'ם שלם לא מוא' הפטוקיאיה.

הפטוקיאיה מבל'ב.

הברפת פרוטוטיפים בפונקציה `void`:

המילה `void` שולבה בו מטאפרה לתפקיד פונקציה המאפשרת לשלוח ערכים לפונקציה הקרויה בלבד. משתי `out` מאפשרות שפה שטחית יותר מערן אחד.

ובודה עם משתנה `out`:

- אין לנו לאפשר את המשתנים הבלתי כמשתני `out`.
- אין אפשרות להשליטם בתוך הפקטרם המוגברם, עד שורש אוותלים במודול בשופט בפונקציה.
- בסיס ריצת הפונקציה, הם ח'יבתם להכליל שורץ כלשהו.

בוגרנו הנה אנו שלחוinos שני מספרים אל פונקציית `mult`, המחשבת את סכום מספרים אלו ומקילום, ומחרורה את התוצאות בערך משוערים שמעוברים כ-`out`.

```
using System;
public class CFunctionExample3
{
    public static void Domath(int a, int b, out int c, out int d)
    {
        c = a + b;
        d = a * b;
    }
}
```

במהלך ריצת התוכנית לאחר ריקירה לפקודת `Swap` עברו א-ז-ע-בפונקציית `Main`.

```
public static int Main(string[] args)
{
    int x = 45, y = 100;
    Console.WriteLine("Before: {0} , {1}", x, y);
    Swap(ref x, ref y);
    Console.WriteLine("After: {0} , {1}", x, y);
    return 0;
}
```

גלו התוכנית:

```
Before: 45 , 100
After: 100 , 45
```

על מנת לאפשר את העברת ערך `by reference` בפונקציה `Domath` נקבעה השמות `a` ו-`b` בפונקציה `Swap` ובפונקציית `Main`. ומשתמת הפקטרם של `Swap` בפונקציה `Domath` שפיה בפונקציית `Swap` נטען `a` ו-`b` שולחים ערך `45` ו-`100` כ-`ref` ו-`ref` ו-`out` שפיה בפונקציית `Domath` שפיה בפונקציית `Swap` נטען `a` ו-`b` שולחים ערך `100` ו-`45` כ-`ref` ו-`ref` ו-`out`.

לאחר הרקומפליאיזציה אין הרבל ב-`ref` הבדל הוא רק בגבולות עלי א-תולן הרטמיים.

- 4 -

גינזאיות חופשית .3.9

ניען לגדיר שעת פונקציית באורה מתקיימת בנקודה x , אם $\lim_{t \rightarrow 0} f(x + t) = f(x)$. מכאן ניתן לראות שפונקציית באורה מתקיימת בנקודה x , אם $f(x)$ שרטוגנאיות בסוג ההפונקטרים שלhorn. במשמעות זו נאמר $f(x)$ שרטוגנאי (stable, ref, ref, ref) במקורה זה, מעתה רקארה לפונקציה f , הפונקטרים שישילו רם אל: שיקבעו איזו פונקציה תופעל.

מהלקה הרא מוגבנה והמאה בהתום פוטנציות ומשגננות מהות שס אוח. המלהקה ראה אוח רהיעאים הפסיסיים ומוחשבים ביחסו בישען #^C, מעסם ריהוה שפה שהה מוכנות עצמיה באוח

רשיין המהלקות הראינו ליקרא את שפת הרובנות למתוחשנות. אנו מתרים עצם ואנחנו געווילו עליות נכל למלתער אעטס מיגרב. אונס (בעורות שיטוט) פועלות כנין היליכ', דבר ונדמה. לאחר שעשגדרט אונס, וועל להבדר למלשל ג'טס, "שְׂקִבָּא אֶת כלא תחתנו עטלת עטאלת אונס" אונס, ווילט עטאלת עטאלת אונס, ווילט עטאלת עטאלת אונס.

מבחן טכני, רמלה השמורה `piothot` את הגדרת המולקה, לאחריה מופיע שם המולקה, ולאחר מכן, בין סוגיהם של סוליטם, מגדרים המשמעות וטוניציות הרשי'ות למלקה.

```
class <class name>
{
}
```

2.01 דוגמא ראשונה

המודול CRectangle מגדיר מחרה בשם `הgeom` מילבן. למחילה שמיינטן פריטים, מופיע פונקציית הופעה על משניים אליו.

```
using System;

public class CFunctionExample4
{
    public static void f1(double d)
    {
        Console.WriteLine("f1(double)");
    }

    public static void f1(int d)
    {
        Console.WriteLine("f1(int)");
    }

    public static int Main(string[] args)
    {
        f1(1);
        f1(1.1);
        return 0;
    }
}
```

IndoorWarrior Project

34 tiny

<http://www.underwar.co.il>

דגם למלולקה:

dagshim labb' hamolaha zemata:
C++: במלולקה הוגדר **Getters/****Setters** כטו שטוקובי בשפה
 ■ במשם המוכר נארה את הדר של שפת C# לשפט **Properties** ו**Setters-i** **Getters**, שרא
 ■ שוה מה שהאנה בדוגמא אונשיות אצת, ומובוות על Properties. C#
 ■ הרטאנו: גון לארות שטהההשאות במשפטים public-i private. מיר
 ■ פונה לוחטוקות עצית יתור בהרשאות אן בראמיים נסחים לב כבר שביבנדו. C++
 ■ אונת מזינט את הדראה לילך כל משינה או שט, ואל בוחן כר בשפה
 ■ C++.

בכט בדוגמא האזהה:
Code מומכתע מהילאה כט שפהורה בדוגמא הדרומה:

```
public class CClassExample2

{
    public static int Main(string[] args)
    {
        CRectangle rec1, rec2;

        rec1 = new Crectangle();
        rec1.setHeight(10);
        rec1.setWidth(20);

        rec2 = rec1;

        Console.WriteLine("Rectangle size: {0} x {1}", rec1.getHeight(), rec1.getWidth());
        return 0;
    }
}
```

dagshim b'molcha:
 ■ הפטואקיה הפונקציה **Main()**, ממנה מתיוללה התוכנית שלוט, אם מבדים משינה מוגה מטלחה שוגה מהילאה (גנבה משינה).
 ■ האזהה ע"פ אובייקטים: כאשר אונת מודרים משינה (גנבה משינה)
 ■ מוגה מטלחה בסיס אובייקט (ב- C#, גון אל מוקאה שבוחן דרכו). לטענה יש לנו בדף משינה הודי מושות (reference) שאים מבעי עלי דבר בעם צירחן, ושלח לבצע פעללה ונסחת עם מה להזקאות ריכוך בשבלו.
 ■ אונ משתחשים באופרטור **new** על מה לילש את המשינה **rec1**.
 ■ גייל זיכין אוטומטי: אין צורך לשחרר את הזיכרון שנטקה בתוכני שטוקאה בשפה **C#**.
 ■ הסבנה שטקה מורה בונת אור בdash אונת שטוקים אינון נשנתיש יוות, מושחרת אונת עברו, ונתן לסט אובייקט סט סקסיפי לטלוליה ע"י השמת שטוק
 ■ גישת שטחים אונת אונד מהתשונם, גם השמי משנתה.
 ■ אונת שטחים באופרטור **(rec1.getHeight();)**.
 ■ גישת שטחים כט שכור און, כי לינש לשורת השונם שי גאנט שטוק שיאז.

```
using System;
public class CRectangle
{
    private int height, width;

    public void SetHeight(int newHeight) { height = newHeight; }

    public void SetWidth(int newWidth) { width = newWidth; }

    public int GetHeight() { return height; }

    public int GetWidth() { return width; }

    public class CClassExample1

    {
        public static int Main(string[] args)
        {
            CRectangle rec1 = new CRectangle();
            rec1.setHeight(10);
            rec1.setWidth(20);
            Console.WriteLine("Rectangle size: {0} x {1}", rec1.getHeight(), rec1.getWidth());
            return 0;
        }
    }
}
```

עbor המשינה לא יצטרם אובייקט **rec2** לא מטרו כי **rec2** שואו.
 באיירה או אונת לא מלהשא כה מהתשונם אל אונת אובייקט.

כאייר אונת אונד מהתשונם, גם השמי משנתה.

Default Constructor .5.01

אם לא מדרים נבר מתחילה כלשיה **C#** ב-**פונקציה** הנקראת בגע שאביב"ק חדש או תקדים רוא את מושגיה. פונקציה זו, איננה מחייבת פרטמים, ומאסף את כל הערכים לאפס, את כל התשומות הבלתי-ל-על ואות כל האובייקטים ל-**false**.
 אם מדרמת פונקציה בוגה כלשיה במקורה, פונקציה זו תרבותה ותפקידו יתאפשר. גישים לב להבדיל בין פונקציה זר לפונקציית ברירת המחדל **ctor**, **C++**, **שאנו** מתחילה את המשמעות בתוליה. רבייה תשב בזורה. אנטימילן משטעים מאפסום, ולאortal אוטם, בזרה זהה צעד חיו עיל מות לשפר את בעשי התכנית.

פונקציות בנות 4.401

פונקציה בוגה מודרנת על ידי שיש המשילה עט סטיין ~ לפע השם.
 פונקציה הוגה מודרנת מתקינה מחרירה ערך, ובן לא **void**, כמו כן, היא אינה יכולה להיות **public** מאוד ליפונציות הרווטות, מכיוון ש-**C++** ש שיש שבות גדרה מתקינה מופע ולא בטוח שפעל בכל מקום נקי היזרין, שהיota התקינה המכיד של הפונקציות הרווטות **C#**-ב-**C++**, אום **C#**-ב-**C++**, והוא געשה באפואו. עקב שבות אלו, חשוב להזכיר הוראות **C#**-ב-**C++**, והפונקציה הוגה מודרנת.

```
public class Crectangle
{
    private int height, width;
    public void SetHeight(int newHeight)
    {
        if (newHeight > 0) height = newHeight;
    }
    public void Setwidth(int newwidth)
    {
        if (newwidth > 0) width = newwidth;
    }
    public int Getheight() { return height; }
    public int Getwidth() { return width; }
    public Crectangle()
    {
        height = width = 10;
    }
}
```

dagma_lipnitska@hotot.org

Properties .7.01

ב-C# ניתן לגדיר משתנים ופוקנציות private->public או private->private. public ו private מגדירים משתנים-> משתנים ופוקנציות->פוקנציות. על מנת לרשט אלי' הינה נחוץ בורר, אך הוליט לזרום מובן דווקא' מושך, והמאפשר לאזט תהליך גוף. מונה זה מוגן ה->properties, שמאפשר לו גם שבודה המוכירה עבורי מוגנה מה->properties. שמאפשר לו גם שבודה המוכירה עבורי רקאה עבודה מלה שודות ישירות, כאשר בפועל, קיראו פוקנציות עם set->get עם קראיה של המשמש.

ספר שרב את המולקה' השם, ושותה בחרביו הוחש. פאג גם כדי הפוקציה Main()

```
using System;
public class CRectangle
{
    private int m_height, m_width;

    public int height
    {
        get
        {
            return m_height;
        }
        set
        {
            if (value > 0) m_height = value;
        }
    }
}
```

dagma_lipnitska@hotot.org

```
using System;
class MyClass
{
    public MyClass()
    {
        Console.WriteLine("MyClass Constructor");
    }

    ~MyClass()
    {
        Console.WriteLine("MyClass Destructor");
    }
}
```

```
public class MainClass
{
    public static void Main()
    {
        MyClass m = new MyClass();
    }
}
```

פלט רוחניים:

```
MyClass Constructor
MyClass Destructor
```

עיר שרב כענבר רוב התיכניות שתוכננו בשיטת C# סבור שדרשו להשתמש בפוקנציות הרוות. הר שבא עם סיבתת GC-הן משמשות ביזור את הגבורה.

בפוקנציות הרוות, מתייחסים בפונקציית הוסטוט מאיר באופן גודל את אזור חמי'ם של

אובייקט שיכל מוגאי'ל את כתמת הזכוכו בפה משתמשת הונגה שילוב ולוכ' משפט עט. בזיאום.

במחלקה נוצרת מVariableBox inherit CRectangle עם היחסות property-m_width וproperty-m_height. המרדרו שיבלהם. היחסים יישמשים בפונקציה get-m_height שsets שנות משנה זו. לאחר מכן שורטט שפונקציית get-m_height יחזיר את תקניתה של ש. שבדקו את תקניתה של ש. היחסים יישמשים באנטיקי. רואו שכלify מ שמשתמש property-ה. מוגנת property-m_width כמ שמתה, אשר כווגת תומולקה הר-property, כי שינית ליבור. בדיקות לפניה שווי הערך.

לאחר הקומפליאציה של הקוד הרצינו מומשו כשי שתוכננו ב-C++, מחד GetPropertyName ווחות selfPropertyName

```
public int width
{
    get
    {
        return m_width;
    }
    set
    {
        if (value > 0) m_width = value;
    }
}

public Crectangle()
{
    m_height = m_width = 10;
}

public class CclassExample2
{
    public static int Main(string[] args)
    {
        Crectangle rec1, rec2;
        rec1 = new Crectangle();
        Console.WriteLine("Rectangle size: {0} x {1}",
            rec1.height, rec1.width);
        rec1.height = 10;
        rec1.width = 20;
        rec2 = rec1;
        Console.WriteLine("Rectangle size: {0} x {1}",
            rec2.height, rec2.width);
        rec2.height = 15;
        Console.WriteLine("Rectangle size: {0} x {1}",
            rec1.height, rec1.width);
        return 0;
    }
}
```

8.8. משתיים ופונקציות סטטיים

```
public class CStaticExample1
{
    public static int Main(string[] args)
    {
        CDoor door1, door2;
        door1 = new CDoor();
        door2 = new CDoor();
        Console.WriteLine("Number of doors is {0}",
```

```
CDoor.DoorCount);
        return 0;
    }
}
```

מש ליב שיכאשר רצין לטעות ל-Property, כינונו DoorCount בשם. כזכור זאת שם המחלקה, ולא שם של מודול אובייקט טביה.

אנטול משותנים סטטיים:

- בירית מהדרי, כאשר און מודלים משתנה סטטי במחלקה, לא מאתהיהם אוטו, רוא מאונטל-ל-0.
- שםמה מפושעת: אתול משותנים סטטיים בגובה פורושת, מודם הגדרתם.

```
public class CStaticExample2
{
    public static int ivariable = 10;
}
```

אם מאתהלים את המשנה הרווט מיד עם הגדרתה. המשנה יאותל פעל את כל במלוא רצונתנו.

- פונקציה בוגה סטטיות: דוד מופת לאותל משלמים סטטיים הרוא על ידי פונקציה. בויה סטטיות, און יכולם לגדיר פונקץיה בוגה כפונקץיה סטטית. פונקץיה זו יכולה לאוותה את המשותנים הרווטים של האובייקט, אך לא את המתוותנים הרווטים של האובייקט. מוגטיה לה לטעות קדילא ליר צירת האובייקט הר阿森 של המלה. הטעות הבוגה בסעיפים עילאה קבל פתרומיטים, אירה מחרה עיר, אך און נינו לגדיר אורה אותה יוציאו און פונקצייתם.

UnderWarrior Project

46 שער

<http://www.underwar.co.il>

UnderWarrior Project

45 שער

<http://www.underwar.co.il>

בבוט בוגרנו הראה, שהרא שמי של המוללה CDoor שארם קווים, המציגו את הכתובת הניל' הינה:

הכאי הפסטן.

In Main()
In Static CDoor Ctor
In CDoor::Ctor

שימו לב: הפקה הסתנית לא תקאה לטעי Main, אלא אף יוצרת האב'קט הואשן.

העברית אובייקטים (עשתי מוחלחות) לפונקציות

כאשר אומרים אובייקט ליטרלית, מערבתת תמיד הדרישות אל האובייקט, ולא ערך של. בדוגמא הבאה: אם אורות מחלקה השמותה מספר (CNumber), (CParamExample), ואפשרות לשנות להקל, אז, אבל אם יוצרים מחלקה שעה (Clock), המפעילה פונקציות שונות רתקבלות כפרמטר אובייקט של המוללה הואשנה.

```
using System;
public class CDoor
{
    private static int m_DoorCount;
    public CDoor()
    {
        m_DoorCount++;
        Console.WriteLine("In CDoor::Ctor");
    }
    static CDoor()
    {
        Console.WriteLine("In Static CDoor Ctor");
    }
    public static int DoorCount
    {
        get { return m_DoorCount; }
    }
}
public class CStaticExample
{
    public static int Main(string[] args)
    {
        Console.WriteLine("In Main()");
        CDoor door1;
        door1 = new CDoor();
        return 0;
    }
}
public class CParamsExample
{
    public void f1(CNumber n)
    {
        n.number = 10;
    }
}
```

01.01 מבנים והעבורה פרמטרים לעתוקיות

מלה: מילוקיות, C#, גם מבנים מבאים לווים מטעמי מתקדם (C/C++) (C) אלים בשפת השתונת C#.

- **C++:** הרබר בין מוחלחות לבניינים הינה בברירת האיש שארם. במלואה רמתה הששה של בירית המודול struct ואילו struct כישנה הינה בברית המודול התה static. public במנויים ובוונתנות או שללים public. הגדרה מוחלטת מודול שארם. מכאן ש'יאר' השפה שארם ש'אילטש לאלטש ל'וואי' קיד שד שונתב בשפת C נעל קומפלירם של שפת C++.
- **C#:** מילוקיות אינן קיימות יותר ב- C#, מכיוון שאם מודדים אותם את היחסאה לפני כל משתמש.
- **C#:** גם בשפתה מודדים מבניה תחביבית למולקלת. כאשר נצער מבנה שותם במלחיה בקומות struct, class ור' מהה שהשת שאיזים יצירם כרגע הא מבנה חדש ולא מולקללה.
- **הבדל בין מבנה למולקללה:** אם כשראות, מודדים מולקללה לעתוקיאה, מעברת תחתיות איניה, לעומת זאת אם עבר ממבנה לפונקציה, וא' ובækש במספר לשארם להנעל התיחסות איניה, היא עותק של המבנה, והוא זהה לש'אילטוקאץ'.
- **רכות גונפית עם מבאים –** העשיה בפרק בתרשף הסוף.

```
public static void f1(ref CNumber n)
{
    n.number = 10;
}

public static void f2(CNumber n)
{
    n.number = 10;
}

public static int Main(string[] args)
{
    CNumber n1, n2;
    n1 = new CNumber(5);
    n2 = new CNumber(5);
    Console.WriteLine("n1 = {0} and n2 = {1}",
        n1.number, n2.number);
    f1(ref n1);
    f2(n2);
    Console.WriteLine("n1 = {0} and n2 = {1}",
        n1.number, n2.number);
    return 0;
}
```

קוד המדגם רואינו עלי מות לראות את ההרבייל בין הערות אובייקט לער. כאשר אנו מעדירים מודולקה לעתוקיאה, התהאה של לרקטיאות הר'א זהה. בשני התקרים, גם כשלקרים ל- f1() וגם כשלקרים ל- f2(), הועברת התיחסות למולקללה לא עותק שלה.

בחוספה המוט בפונקציה, יוזפו י' גם ח' שום ל- 10.

ניר אדר
nir@underwar.co.il

דגם:

```
public class CstringExample1
{
    public static int Main(string[] args)
    {
        string s1 = "Hello";
        char ch = s1[2]; // Ok
        s1[2] = 'x'; // Error
    }
}
```

אם ייכילם ללבול את חוו המחרחת, אך איןנו יכולים לשוטר אותן.

למהירהן פונקציית `string::operator+=` מפעילה הרבה יותר מאשר `string::insert`?
 המחרחת יצטרח למעשיה פונקציה נוספת שמשתמשה בתוללתה השיטה `operator+=`, ולא משנים את המחרחות עצמה.

דוגמא: הפעוציה `string::insert` קיימת מחרחות חדשות, אבל המחרחת חדשה עשויה לא ליצור מחרחות חדשות, ולכן השיטה `operator+=` מושגנת.

מיודמת כי `operator+=` את העבודה עם המחראות –
 שפת C# תומכת בה אעד נספּה מודר טריסטו יוזם וושה מודר לזרור אחורית מחראות –
 המחראות הרשותה ב-`C#` מילאה תפקיד `string`-ב-`UNICODE` מילאה תפקיד `string`-ב-`ASCII`.
 השפטות נתקיימות בטלום.

1.11 מבוא ותביב'

מחרחות היא אובייקט של תype סטנדרטי.

בשפת C ממעשו מחרחות על ידי מערך של תype טוים, או אטשרה שורה מלאה של ספריות לחראות כל מעכלי תווים, או אטשרה שורה מלאה של ספריות לחראות כל מחראות.

מחרחות הן אנד הולוקרים החברים של השפה – מיין שיטות מחרחות אונטימיים מביניהם אוטן ייאו את העבדה עם המחראות.

כל אחות מחרחות מוחדרת לשבובת של המחראות, וכן המדרירה בשפה תספורת אוטן ייאו מחראות יצירות מיילם, משפטים והודעות שילוגות בגע אוטן.

מחרחות כ-`string` מחרה עצמה עד נספּה קידמה, וושה מודר טריסטו יוזם וושה מודר לזרור אחורית מחראות –
 המחראות הרשותה ב-`C#` מילאה תפקיד `string`-ב-`UNICODE` מילאה תפקיד `string`-ב-`ASCII`.

דוגמא להגדלת מחרחות:

```
public class CStringExample1
{
    public static int Main(string[] args)
    {
        string s1 = "Hello";
        string s2 = "World";
    }
}
```

מחרחות הן שארם מקצועים של המחרחות, ואילו החרמתה. נאנו יייצרים מחרחות, מctr למשהה אובייקט שמשתמש במלמד שאותו לא מחרה, מctr משגהה המחרחות `s1`, למישר, רוא משגהה המחרחות `s1`, `string::operator+=` מילקוט שוגג המיל את המחרחות `"Hello"` וראת. הסיפוי הוא מחרה המשתמש במלמד `s1`.

אחרי החקיקה לתוכניאה `Insert`, המחרה `s1` משיכלה לחרה במלמד המקרים, ואילו המחרה `s2` תהייך במחרה `"Hello World"`.

נימתו להשתמש בו כאזורנוות + יול מינית לאלכטרה בווי אטומיזציה

StringBuilder .2.11

על מנת להשתמש ב מחורת System.Text |, עצור לזרוק את המחרה \Console.WriteLine |.

```
using System;  
  
public class CStringExample1
```

```
using System;
using System.Text;

public class CStringExample
{
    public static int Main(string[] args)
    {
        string str = "Hello";
        StringBuilder sb1 = new StringBuilder();
        Console.WriteLine(sb1);
        sb1[1] = 'Z';
        Console.WriteLine(sb1);
        return 0;
    }
}
```

- **עדרות אלין** תשלב לשיטות ב-`System.StringBuilder` עם `System.Text Namespace` מבעוד שבסירה השניה בהן).
 - **יעזרת העברת** בשוקצתה הבונה מהורת גלילה. תמשוחת רתמי.
 - **המחלקה** של `StringBuilder`-ה אודא לו של האבר הראשי במתחרות היה 0. כאשר שינינו את הרתא `sb[1]` התא השוי במתחרות השועף.

3.11. מחרוזות ותווים מיוחדים

בפרק 5 במאמר הנה ניתן לראות רשותה מלאה של הרויטס המוחדים המשרתים כאריך או רואץ להשתמש בתם, "על מנת להברור לרשותה שמדובר בסימן ↷, כמ"כ-ב-*C/C++*, שם אוקט הוויס מוחדים, שמותר לרשמה של רשותן ↷.

LITERATUR

C# מאפשרת לנו ליצור מתקשרות של נתונים, אם אנחנו יוזם שברוחות לא הילכים להוציא נתונים מתחום מסוים @ לפי המחרחת, ואומרים בוכן לקובץ פילר להעתם מהותי המודפס.

וְאֶלְעָזָר הַקֹּדֶשׁ (ל' שְׁמִינִית אֲבָדָה) וְאֶת הַדָּאָסָן, עַם).

- **ריקו המחרה** – כאשר הדרות את האור של המחרה ל-0, ריקו יעשה את **Console.WriteLine**.
- **Console AppendFormat** – כאשר קראנו ליטופת ה-**StringBuilder AppendFormat**, אוטומטיים.
- **Console.WriteLine** – בפונקציה **StringBuilder ToString** שאותו שמשים כבפונקציה **StringBuilder.ToString()**, בפומט זהה שאותו שמשים כבפונקציה **String.ToString()**.

נובעים בדמיון נזקף:

```
using System;
using System.Text;
public class CStringExample1
```

```
public static int Main(string[] args)
{

```

```
string s1 = "Hello";
int i = 10, j = 20;
StringBuilder sb1 = new StringBuilder(s1);
Console.WriteLine(sb1);
sb1.Length = 0;
sb1.AppendFormat("({0}, {1})", i, j);
Console.WriteLine(sb1);
return 0;
```

כלי תומך של `StringBuilder` שהכרת בדף זה:

AppendFormat: כאשר קגנו לנקז אפס, AppendFormat מוחזרת./class="list-item">-

.Console-ה של WriteLine

תורה

UnderWarrior Project

56 tiny

<http://www.UnderWar.co.il>

卷之三

55 tiny

<http://www.underwater.co.il>

בדוגמא זו אם רואים את הדריכים השונים להקצת זיכרון למעריך:

- **C#** מודול, ושאר מודולים יוכנסו לתוכה.
 - **arg3** מוגדר, ומשהו לאחר מכן אנו מזקדים שבו יוצג.
 - **arg4** מוגדר, ושאר מודולים יוכנסו לתוכה.
 - **arg5**, אליהם מזקדים עבור מיפויים עובה זעירו.
 - **arg6**, אליהם מזקדים עבור מיפויים עובה זעירו.
 - **arg7**, אליהם מזקדים עבור מיפויים עובה זעירו.
 - **arg8**, אליהם מזקדים עבור מיפויים עובה זעירו.
 - **arg9**, אליהם מזקדים עבור מיפויים עובה זעירו.
 - **arg10**, אליהם מזקדים עבור מיפויים עובה זעירו.

הנחיות דינמיות ראשונות

卷之三

ב-1.1. בנה. המער מוחמש עבור כל סוג של המערךת, ובעור סוגים לא מומומשים מוצרת

מתקנה המאפשרת עבור הוגג אוטומטי, ולעומת זאת ב-**check**.
והלא התרחש שמיים לאזרר הפיכת C-R לגורן לאזרר הסופט, סמלות של גוריות
לשלוט, וונדר הרפ' לבניי דנדמי, יישון סוג אחד של מושך עבור כל סוג הערכות.

```
int[] arr1;
long[] arr2; arr3;
```

בוגם זו הגדתו 3 מערכם.

ב' ט' ט' ט' ט'

1.2.2. גָּלוּעַת מְעֻרִיכִים לִפְנֵי הַצִּיָּה

ונס לבר שסבירו מאטחיהם מעוד לנודו מוסים אינו חיבם להשתמש בקבוע מוחות לנצח לשם משונה סיכיל את גודל המבוקש של המערר.

מערלים מושברים ליטופאצית על ידי "היתותם (reference)" בברית המוחרי, שכן הפעונאים מתקבלים לתמעשה את המשער ואלה העתק של.

מפקת מספר פתרונות אחרים על מנת לגלוות את גודל המשגר אמורים נראות מ-^{א'}

3.21. גונרטה המהירת מער

פונקציה יסילה גם להחזר מערך. מעילים ב-C# מזקאים על העירמה ולא המונוט, ולכן

```
int[] arr3 = new int[] {1, 2, 3, 4, 5};  
int[] arr4 = { 1, 2, 3, 4, 5 };  
  
arr2 = new int[20];
```

UnderWarrior Project

100

UnderWarrior Project

58

<http://www.underwar.co.il>

```
return 0;
```

```
}
```

כדי בפונקציית השוואה.

- (PrintArray() ו-foreach שברות עלי האברים תבעה).
 - מתקנית מערך, ובהתאם הופקאה של האיברים שער. הפתרון המשך `ChargeArray()`, שברת על כל איברי המערך, במעבר ההפוכה של האיברים שער. הפתרון המשך `GetLength()` ו-`0`.
 - ראה שב את פונקציה זו משעיה ופונקציה מערך מודים.
 - הפונקציה `GetLength()` ו-`0`.
 - בוגר ל-C/C++, פונקציות יפות להזין מערך. המערך מושך להזינה `ReturnArray()` כ-`0`.
 - מאוחר שהפונקציה `ReturnArray()` תחזיר.

בפונקציה `PrintArr()` ובכל, ניתן לieżע על כל מערך לוולא מושם שמחזור גלגול ערך (`IEnumerable<T>`) מוכן (`System.Array`). יושת את התשassed במחזור מערך ועל ידי לאפשר מבחנה בקרה זה ואחרים.

```
using System;
```

```
public class CarreysExample1
{
    // The function gets an array and print it
    public static void PrintArray(int[] arr)
    {
        foreach(int iValue in arr)
        {
            Console.WriteLine("{0} ", iValue);
        }
    }
}
```

// The function gets an array and adds 2 to each element in

```
// the array
public static void ChangeArray(int[] arr)
{
    for (int i = 0; i < arr.GetLength(0); arr[i+1] += 2);
}
```

// The function creates new array and returns it

```
public static int[] ReturnArray()
{
    int[] arr = new int[]{1, 2, 3, 4, 5};
    return arr;
}
```

// Main Program

```
public static int Main(string[] args)
{
    int[] arr1 = new int[10]{10, 9, 8, 7, 6, 5, 4, 3, 2, 1};
    PrintArray(arr1);
    ChangeArray(arr1);
    PrintArray(arr1);
    int[] arr2;
    arr2 = ReturnArray();
    PrintArray(arr2);
}
```

jagged מערך 2.4.21

מערך **jagged** הוא למשה מערך חד ממד', בו כל איבר הוא מערך בפוי עצום. אassetת לפקח את המערך, ואחר מכן תקח אותה כל איבר ב-בفرد.

דוגמא:

```
using System;

public class ArraysExample2
{
    // Main program

    public static int Main(string[] args)
    {
        int[,] arr;
        arr = new int[4][];
        // four rows in the array
        arr[0] = new int[10];
        // 10 cells in the first line
        arr[1] = new int[7];
        // 7 cells in the second line
        arr[2] = new int[3];
        // 3 cells in the third line
        arr[3] = new int[24];
        // 24 cells in the last line
        arr[3][2] = 2;
        // access an element
        return 0;
    }
}
```

מערכים רבי ממדים 1.4.21

ב-C# שנו סדרם של מערכות רבי ממדים - מערכות מלבניות: **multidimensional arrays**. מלבניים הם מערכות בהם כל השורות באותו אורך, אליו מעריכים מלבנים שורות שונות יובילו לרווח שווה.

הגדלת מערך רבו ממדים מחייבת בשורת ר' ישום סיסקיים בתרבות הסוגרים המהובגים, כירה מהקה לרשota את כל הממדים של המערך, למשל:

```
int[,] mat;
int[,] cube;
```

הקצתה המעריכים תששה באורה הבהא:

```
mat = new int[10,5];
cube = new int[3,3,3];
```

ישה לאיברים במערך זו ממד' תשעה בזורה הבהא:

```
mat[2,3] = 3;
cube[1,2,1] = 4;
```

5.5. שיטות וmethods של מערכם

Clone(): שיטה פונקציה זו יוצרת רעתק של המערך ומוחירה אותו.
תאורה: מוחירה למספר שיטות (Methods) ומאפשרים, בתם או נילים (Methods), בינם לבין המשמש.

```
int[] arr1 = { 1, 2, 3, 4, 5, 6, 7, 8 };
int[] arr2 = (int[])arr1.Clone();
```

Rank: מאזין המהיר את כמות המומדים של המערך.

תאורה: מאזין המהיר את כמות המומדים של המערך.

Length: מאפיין מוחיר את מודר המערך, שהוא מכטלה אובייקט כל המומדים שלו.

תאורה: מוחיר את מודר המערך, שהוא מכטלה אובייקט כל המומדים שלו.

Sort(): שיטה דוגמא:

```
int[] arr = { 7, 5, 3, 4, 1, 9, 2, 8 };
System.Array.Sort(arr);
for (int i = 0; i < arr.Length; ++i)
    Console.WriteLine("({0}) ", arr[i]);
Console.WriteLine();
```

GetLength(): שיטה דוגמא:

```
int[,] cuude;
cuude = new int[7, 8, 9];
Console.WriteLine(cuude.Rank);
```

הערך שודפס יהיה .3.

Clear(): שיטה דוגמא:

int[,] cuude;
cuude = new int[10, 20, 30];
Console.WriteLine(cuude.Length);

הערך שודפס יהיה .10 20 30=6000.

```
int[,] mat;
mat = new int[10, 20];
Console.WriteLine("GetLength(0) = {0} and GetLength(1) = {1}",
    mat.GetLength(0), mat.GetLength(1));
```

GetLength(): שיטה דוגמא:

```
int[] arr = { 1, 2, 3, 4, 5, 6, 7, 8 };
System.Array.Clear(arr, 2, 3);
for (int i = 0; i < arr.Length; ++i) Console.WriteLine("({0}) ", arr[i]);
Console.WriteLine();
```

תאורה: פונקצייה המאפשרת תוספות ומחיקות ערכים שונים בתום שיטות שבר. ומוחילה לnull.

SetText(): שיטה דוגמא:

```
GetLength(0) = 10 and GetLength(1) = 20
```

64 טען

<http://www.underwar.co.il>

[UnderWarrior Project](#)

63 טען

<http://www.underwar.co.il>

[UnderWarrior Project](#)

3.1 מבנים תחביריים וספירים בשפה

שיטה: `IndexOf()`

תאונה: ההפונקציה מוחקיה את האותיות שירשוף הרាសן של שער מסום. הטעוקצתה מוביל שר כפטור, ומחרירה את אידידטי של האבר, וא-1- אם לא נזאת איבר סודה. יתקייל פגיעה זו רק על שריכם ח'ר מז"מ'.

דוגמא:

```
int[] arr = { 1, 4, 56, 23, 2, 43, 23, 12, 83 };
Console.WriteLine(System.Array.IndexOf(arr, 43));
```

כ"ש שבד ראם, מיל מלבנים (structs) בנוסח למולקלוקות, כאשר מנבירים מהחקלא לטעוקצתה, שבסורת תחומי ההיי-יחסות אליה והעומת אותה, אם מנביר מבנה להטוקצתה, וכן במספרה להיבור התיחסות אלוי. "אייר שוגה של המבנה, והוא זה שיעגבו לאטוקצתה, מבנה בעל סמלטיות שאלים. ואה מזקה על הרהטת האל עלי העיינה, בתוספ- –

הא קדים בסהוף של הטעוקצתה מהא מודור – כששוחה הטעוקצתה מוסימת את פועללה, מספיק המבנה יתתקזק.

1.31 מבנים

דגם למבנים.

- מול המבנה, במונים מוחלץ לשימושה ב内幕 רג'ו אובייקטים עם מוט נוימים.
- מיקומוטיפ מתייצם שבאותן אידיאלי גודלו שי' struct לא יורה מעבר-ל- bytes¹⁶.
- פוקאציות בוגרת לא ניתן לשוא מבנה בעל אי-פרטמיים. און כילים למתוך הן באנס עם פתרם. הקומ"ל יוסי פ. ג' פ. מורה באירוע מוחלט ביל' פומטרם. ש. איסאקס את כל שורת המבנה.
- הורשה: ב黑马קס וואה מהחולקות הנורשות מתחלקות אחרות. מונים לעתות מוחלקות אים יקרים לשעת מוגנים או מחלקות אחרות. כמו קהם אים יקרים את און האביסיס למלוקות.
- פוקאציות הורשות: לא ניתן לנתח פוקאציה הרוותה (destructor) נבור מבנים.
- לאורה רהה, סגול לחיות תחון בdishorthush (מכיוון שעוד בפונקציית הורשתם עם מבנה, מכיוון שעוד בראק מובנה הורסת (סוק הפקה בה רהה ונתן לעשרות זכרון דטרמיניסטי לתוכו). עם זאת, שפת # בוהה שלא לאפשר צורה פוקאצית הורשות בעבור מבנים – סויין לשות ר' יגור שגייאת קומפלקסית.

```
int[] arr = { 1, 34, 45, 56, 61, 78, 88, 89, 2037 };
Console.WriteLine(System.Array.BinarySearch(arr, 61));
```

שיטה: `Binary Search()`

תאונה: פגאייה זו מחפשת שור במער, לי' אלגוריתם של חישוב באנא. על המערך ליהו מתיין.

דוגמא:

- צוואר מוגן: ניתן לאזרז מוגן ללא שום באפרור השם בתקرار כה, וכך, מוכן לסתמש במוגן רק אחר שאתוחל את כל השודות שבע.

```
class MainClass
{
    public static void Main()
    {
        // Initialize:
        Point myPoint = new Point();
        Point yourPoint = new Point(10,10);

        // Display results:
        Console.WriteLine("My Point: " + myPoint);
        Console.WriteLine("x = {0}, y = {1}", myPoint.x,
                          myPoint.y);

        Console.WriteLine("Your Point: ");
        Console.WriteLine("x = {0}, y = {1}", yourPoint.x,
                          yourPoint.y);
    }
}
```

התוצאות תראה:

My Point: x = 0,y = 0
 Your Point: x = 10,y = 10

```
struct test
{
    public int i;
}

public class CStructExample
{
    public static int Main(string[] args)
    {
        test t;
        t.i = 3;
        Console.WriteLine(t.i);
        return 0;
    }
}
```

תגמול איז, אם נמוך את השרה המודגשת, נקל שיאת קומפייציה.

תגמול לשימוש במוגן:

```
using System;
public struct Point
{
    public int x, y;

    public Point(int px, int py)
    {
        x = px;
        y = py;
    }
}
```

כמ' Ci, ובוט בדוגמה הינה:

```
using System;
class MainClass
{
    enum colors { BLACK, WHITE, RED, BLUE };

    public static int Main()
    {
        colors i = colors.WHITE;
        Console.WriteLine("101", i);
        return 0;
    }
}
```

באורuml המייל השמוו אונס כל מה שאל קביעם.
המשמעות הבסיסי דוגמה לשוטה:

```
using System;
public class EnumTest
{
    enum Days { Sun = 1, Mon, Tue, Wed, Thu, Fri, Sat};

    public static int Main()
    {
        int x = (int) Days.Sun;
        int y = (int) Days.Fri;
        Console.WriteLine("Sun = {0}", x);
        Console.WriteLine("Fri = {0}", y);
        return 0;
    }
}
```

על המור וופט המילה **int**, ולא **WHITE** או **BLACK**.
כברית מודול,enum מוחק ביצירן כמשתנה מיפוי של **int**. עם כל הערך אונס של
הוא אופס ערך של כל קבוע באחד. וויק לישתו גם את סוג המשנה וגם את
ערכיו הקבועים:

```
enum grades : short { FAIL = 40, PASSED = 55, EXCELLENT = 100 };
```

מורת לפי השיטה נמצאת מתקה הירשות את System.Enum ומילה מספר שרים
Enum, הגמירה ללהות נועשת בסיס את שאר והמרת מעור בחרה בזעם.
ובעם, מוכנע באמצעות מיזן (Hash Table) אשר מוחול במקורה.

Enumerations .2.31

readonly.4.31

גדרת קבועים בחרות המילה **const** ב-C#-ש תגינה, חובה **readonly** משונה מהדריך **readonly**-הו. ואחר מכך יופיע ר'ן **readonly** משווה שמיון או במאים המתוארים או – אמתן עם צוות האבטחה, ואותה מכוון שמייה על קבעה. תגינה שמיון משמשת תריסים רבים מהמאים המתוארים או – אמתן עם צוות האבטחה, ואחר מכך שמייה על קבעה. קבעו אונט – במידה ונוסף לשוטה בשורת שם המולקה. שורות לו באיתור באים – במידה ונוסף לשוטה בשורת שם המולקה. נבל הולא שאגא, מתה מושג **readonly** לא-אביק, ולא למולקה.

דוגמה:

```
using System;

class Person
{
    public readonly string Name;
}

public Person(string szName)
{
    Name = szName;
}
```

כגמ"ס.3.31

גדרת קבועים בחרות המילה **const** ב-C#-ש נישית בחרות המילה **const** שמיון במלוקה כל קבעה כשלשה, קבעה ערך שאמן משווה במולך לרשותה החביבית. כל קבעה כשלשה, וגיאשה להקבע גנשטי בשורת שם המולקה. קבעו אונט – במידה ונוסף לשוטה של המולקה.

דוגמה:

```
using System;
class MathClass
{
    public const double PI = 3.142;
}
```

השם שמיון לקובועים הם לאכלי קיריאות הקוד (שםם משמשתיהם לעכיפם) ומצרכי הגדאות כלילו הักษורת למולקה.

```
class MainClass
{
    public static int Main()
    {
        Console.WriteLine("PI = {0}", MathClass.PI);
        return 0;
    }
}
```

השם שמיון לקובועים הם לאכלי קיריאות הקוד (שםם משמשתיהם לעכיפם) ומצרכי הגדאות כלילו הักษורת למולקה.

```
class MainClass
{
    public static int Main()
    {
        Person perl = new Person("Moshe");
        Console.WriteLine("Person name is {0}", perl.Name);

        // The following line won't pass compiler:
        // perl.Name = "David";
        return 0;
    }
}
```

6.6.31. הוראות למחדר

Boxing, Unboxing .5.31

וין ליתר ב-#include זה או מהדר על מנת לבוא קומpile'יה מותנית.

הפקודות הן דומות לאלו של C/C++'ל, מבחינה כתובתה וכמהן הינם המשמעות, מלבד הטענה שלא

שעלית נעשית על ידי השם משנתה ערך למשנתה ערך'ל precompiler או ה-#include'יה נעל'ל.

#define, #undef, #if, #else, #endif הפקודות הן

CPoint p1 = new CPoint();

object o = p1;

וגם:

בוגר נכל לרשמה שבס-ב-#error יונת להתריע על ביצוע.
וגם:
כ-לי להפוך משנתה התייחסות לשער'ו, ומשנתה הרחבה לשער'ו.

:Casting-ב-משנתה התייחסות לשער'ו, ומשנתה הרחבה לשער'ו.

```
#if ! SIMULATOR
#warning No Simulator defined
#endif
```

```
CPoint p = (CPoint)o;
```

- boxing ו-unboxing (פונקציית רישום ופונקציית גירוש) בפערם של הרכבים הקיימים בזיכרון.

מקיון שהרטה המשמר התיקתי שומר על השער'ו, כאשר מזיא שר' מרשמה (על מנת לגשת לעור' עצמן, עצור לבקש מהר' ליטיפיו מה מהימן), ובמקרים מסוימים.

חשיבותם של שפעולות אלו היהות – המלה אל מובצעות כאותם, אלא הסברבה מעוצעת את פועלן ה-ה-boxing/unboxing. וכךואה מכך חלק משגיאות המהא'קסים יתחל' רק בדרכו ריאצ'ה. בוגר' נעל'ם בתר' אמרע' לעיל' מוכגנו'ו, בסה' למינ'ה את כמות הפעוט' שפעולות אלו מתרחש'ת.

2. דוגמא ואשונה

1.41 הורשה .41

C# הורשה בשמנה 1.41

```
using System;
class CPoint
{
    public int x, y;
}
class CPoint3D : CPoint
{
    public int z;
}

class MainClass
{
    public static int Main()
    {
        CPoint3D myPoint = new CPoint3D();
        myPoint.x = 3;
        myPoint.z = 10;
        return 0;
    }
}
```

השאלה בז' מונען הרושה של **C#** לטענו הרושה של **C++** שפות שפותם בחלוקת הירושה ממלוקות אחרות. ממלוקה הירושה ממלוקה אחרת מבלוטת כל הרכונות של הממלוקה הממלוקה הירושה ממלוקה ומולוקה אותה שפה את **C#** שפה את **C++**. ב**C#**-ל-**C**, מחרת הורשה ממלוקה אותה בבלוט (אין חורה ממלוקה), וכן מילויים. ביגוד-

השאלה בז' מונען הרושה של **C#** לטענו הרושה של **C++** שפות שפותם מארה

ב**C#** כל ההורשתה הן מוגן **public** לשומות.

- כאותם בשמנה **C#** הרושה ממלוקה ממלוקה.
- כוון לעבעתו שמיוסד **interface** קע במחסן.

ממלוקה הנרשמת ממלוקה אחרה ממלוקה את כל הטענה. הזרה והלילית של הדרת ממלוקה הנוראה אורה ראות:

מודרם ממלוקה בשם **base_class_name**

מודרם ממלוקה בשם **derived_class_name**

מודרם ממלוקה בשם **base_class_name**

3.41. גישה אל שדרות הבסיס

מחלקה הינה שורה אחת, יכולה לארח, יכולה לרשום public-ה שליה, אך אל שורת private-ה.

כשэр או יצרים אובייקט חדש, שנדר לאktor מסוים אוו, ונצה לתקן לפרטיה הגונה של האובייקט החדש. על מנת לעשות זאת, משתמש במשורה base, המתייחס אל המולקה ממנה נוצרת המולקה (בנעד-ל-*C*#-*C++*, *C*-*C++*, שבסה-ו-*C#*).

כך מושתמש במלחילה השמורה base עם מות לתשעת כל הפקציות של המולקה ממנה גורו.

כדי לומר כי פונקציה במלחילה הנגדודה מוחילפה במלחילה המקורי, ישמשם בפותרו *new*, ביט בדגמא:

בביס בדגמא הראתה:

```
class x
{
    private int i;
    public int GetI() { return i; }
}

class y : x
{
    public void f()
    {
        GetI();
        / / i = 3;
    }
}
```

מחלקה עירשה את המשמעות ו- *אאות* הפקצת *x*.
אם אתה רואים יכולת לטעות לך איך רתוקיצה .GetI().
פירות אל המשמעות ו- *ונורם* לשלב קומפלקס, מיפוי שואה משוגנה פרטישל *x*.

```
using System;
class Rectangle
{
    private int m_x, m_y;

    public Rectangle(int x, int y)
    {
        m_x = x;
        m_y = y;
    }

    public int GetX() { return m_x; }
    public void SetX(int new_x) { m_x = new_x; }

    public int GetY() { return m_y; }
    public void SetY(int new_y) { m_y = new_y; }

    public int GetArea() { return m_x * m_y; }
}

class Square : Rectangle
{
    public Square(int len) : base(len, len)
    {

        new public void SetX(int new_x)
```

Sealed Class .6.41

שפת C# מאפשרת שאליה יינה לגור מחריקות חדשות. מחריקה שאליה גוראות ממנה קוראות מחריקה מתויה. **sealed** מגדירה מחריקה המתויה נישת בטענה המלאה שהטוויה

לדוגמא:

```
sealed class MyClass
{
    public void SetX(int new_x)
    {
        base.SetX(new_x);
        base.SetY(new_y);
    }
}

class MainClass
{
    public static int Main()
    {
        Square r1 = new Square(4);
        Console.WriteLine(r1.GetArea());
        r1.SetY(5);
        Console.WriteLine(r1.GetArea());
        return 0;
    }
}
```

protected . הרשאה 5.41

משתנים או פונקציות מהקבילים הרשאה מוגןים, בדומה לשפתת C++, נישם במחלקה בה רום הוגדר, במחלקה הגוראה ממנה אין לא מוחך אליהם. אולם משתנים בהרשותה זו כמו שאנו משתמשים – **public** או **private** – פשוט אמורים לעסוף ראיון רצויים לסתורו הרשאה זו.

לדוגמא:

```
protected int i;
```

גילה מורה ים

۵

```
public int GetArea() { return m_x * m_y }

}

class Square : Rectangle

{
    public Square(int len) : base(len, len)
}
```

ט. גאנז. האזרע

רבות רואין את התמצות הרחיקות של שפת הננות מוחות עמיים.

```
public override void SetY(int newY)
{
    SetX(newX);
    base.SetY(newY);
}

virtual void SetX(int newX)
{
    base.SetX(newX);
    base.SetY(newY);
}
```

—

```
public static int main()
{
    Square r1 = new Square(4);
    Console.WriteLine(r1.GetArea());
    r1.SetX(5);
    Console.WriteLine(r1.GetArea());
    SetLengthTo7(r1);
    Console.WriteLine(r1.GetArea());
    return 0;
}

public static void SetLengthTo7(Rectangle r)
{
    r.setX(7);
}
```

LIXXV:

```
using System;

class Rectangle

{
    private int m_x, m_y;

    public Rectangle(int x, int y)
    {
        m_x = x;
        m_y = y;
    }

    public int GetX() { return m_x; }

    public virtual void SetX(int new_x) { m_x = new_x; }

    public int GetY() { return m_y; }

    public virtual void SetY(int new_y) { m_y = new_y; }
}
```

UnderWarrior Project

82 tiny

```
Vector v1 = new Vector();
v1.x = 10;
v1.y = 5;
```

```
Console.WriteLine("Vector Length: {0}", v1.length);
Vector3D v2 = new Vector3D();
v2.x = 0;
v2.y = 1;
v2.z = 1;
Console.WriteLine("Vector Length: {0}", v2.length);
return 0;
}
```

המשתנים בדוגמאם גולרים, ומכאן מוחלט כי מטרת שחרר אותו מוחלט, וע"ל מלה לשפט את הנדר שוקטור דן מהר, לאחזר מכך ונפוץ אותו למלחת מודר.

```
using System;
class Vector
{
    public int x, y;
    public virtual double length
    {
        get
        {
            return System.Math.Sqrt(x*x + y*y);
        }
    }
}
```

לעתים נוהג לישר מוחלקיות, שמשמעותו לאור גירה בלבד, ולא ליערת אובייקטים שלין, מוחלקיות אליל וקளות אבסולוטיות. ג'אג בדורות הופוצאה, שהמשתמש בשונה רה אבストראקטית לאן לאגדיר או לא נתקל בה להקמתם מוגנה.

```
using System;
abstract class Food
{
    public virtual void eat() 
    {
        class Banana : Food
        {
            public int z;
            public override double length
            {
                get
                {
                    return System.Math.Sqrt((base.length*base.length) +
                        z*z);
                }
            }
        }
    }
}
class MainClass
{
    public static int Main()
    {
        UnderWarrior Project
        http://www.underwar.co.il
    }
}
```

```

class MyDerivedC: MyBaseC
{
    public override void MyMethod()
    {
        x++;
        y++;
    }

    public override int GetX // overriding property
    {
        get
        {
            return x+10;
        }
    }

    public override int GetY // overriding property
    {
        get
        {
            return y+10;
        }
    }
}

public static void Main()
{
    MyDerivedC mC = new MyDerivedC();
    mC.MyMethod();
    Console.WriteLine("x = {0}, y = {1}", mC.GetX, mC.GetY);
}

```

מזהה מיליה המשורה לפי הדרישה, כאשר אם מגדירים טווקיאט כמשמעותם או מדים את שמה אז רואים אותו, מוגדרת מוחלטת אבסולוטית, כי בודת שהשכל בחלק override עד לאכוף את הפקטיות האבסולוטית, כמו ק' אונטראיאט properties דן ואבסטראקטווער.

דגם:

```

using System;
abstract class MyBaseC // Abstract class
{
    protected int x = 100;
    protected int y = 150;
    public abstract void MyMethod(); // Abstract method
    public abstract int GetX // Abstract property
    {
        get;
    }

    public abstract int GetY // Abstract property
    {
        get;
    }
}

```

• פונקציה זו מבצעת השוואת בין אובייקטים. כבירות הוחזר.

- הfonקציית השוואת התהייה שולץ בלבם, כמו זה מחייב שיבר את המשנים מאבעים אל אותו אובייקט.

דוגמא:

```
using System;

class MyClass
{
}

class MainClass
{
    public static int Main()
    {
        MyClass c = new MyClass();
        MyClass c2 = c;
        MyClass c3 = new MyClass();
        Console.WriteLine(c.Equals(c2));
        Console.WriteLine(c2.Equals(c3));
        return 0;
    }
}
```

כפ' שראנו, ניתן לארבע על מנת לחקות בסיס, על מנת לאבדע אל אובייקטים של מחלקות הגזירות ממנה.

אם נוצרת תהיתם למשתנים פותחנאות של רומחילקה הנגררת, וכל לבער המורה בחרה אל הטיפוס המקרה של המתלהקה, בעורות .casting. אחר שנעשה המורה, תוכלSab לשאל המשנים והטיפות של המהילה הגדולה. אם רואו אובייקט בפועל, הוא לא מתרטט שאל המורה, 'של' exception thrown.

ו'ין לבודק אם משנה הואה סוג מסוים או אם יש אותו של שום בתבילה השווה :is

```
MyDerivedC c = new MyDerivedC();
if (!c is MyBaseC)
    Console.WriteLine("You will never see this message");
```

object פונקציות של override יב. 4.51

כל המחלקות ב-C#-ן מודרניות מוחלפת הרשות .object בינהן אוטומטית. שינון להגדירן מוחש במחלקה המוגדרת.

המחלקה של :

```
objectօbjecτ{
    public string ToString();
    public virtual int GetHashCode();
    public virtual bool Equals(object o);
    public virtual Type GetType();
}
```

- פונקציה המוחזקת בין ערכי התהווות של שניים – **object.ReferenceEqual**.
- אובייקטים.

- פונקציה המוחזקת בין מחריהם מודג'ים – **ToString**.
- פונקאה הממבלטת שניים – **GetHashCode**.

לחיות "זה" כל שילא יחו שיבן אובייקטים בניי, אותו מפרק את האובייקט הספציפי. המספר ציר משמש להוכנסת האובייקטים למובנה מסוג Hash Table. אם מודרים מוחש את הטעקיה רלווד'ר גם פונקציה זו.

Casting .3.51

כפ' שראנו, ניתן לארבע על מנת לחקות בסיס, על מנת לאבדע אל

אובייקטים של מחלקות הגזירות ממנה.

אם נוצרת תהיתם למשתנים פותחנאות של רומחילקה הנגררת,

הטיפוס המקרה של המתלהקה, בעורות .casting.

אחר שנעשה המורה, תוכל Sab לשאל המשנים והטיפות של המהילה הגדולה.

אם רואו אובייקט בפועל, הוא לא מתרטט שאל המורה, 'של'

דוגמא:

```
MyDerivedC c = new MyDerivedC();
if (!c is MyBaseC)
    Console.WriteLine("You will never see this message");
```

Interface .61

Namespace .1.61

בתוכית גדרלי, יתנו התנשיות בין שמות מחלקות שונות שאישים באתות מפוחמים. בעוגה גנרי ניתן לפתרו את הבעיה זו. מגד "מ robab Shmota", שיעוט את המחלקות, וירה תילק מיחסם המלא של המחלקה. דוגמא:

```
class MyClass
{
}

class MainClass
{
    public static int Main()
    {
        MyClass myobj = new MyClass();
        return 0;
    }
}
```

המילה `using` אמורה למודר ליחס שמו של מחלקה/ת מבנים בתוך ה-namespace. כשהוא בא לתייחס שם מחלקה באות שמי' לא. מושג בעה, מכיוון שהיא יונק לפות מהשם של המחלקה. מוגדר בעה, namespace בנה מחלקה כזו, הינה השם למחלקה. מוגדר גי' רישום ה-namespace-ה החיצוני והפנימי, ואחריהם שם המחלקה, כשתה מפדרים בתוקנות בינוות.

נית לנו לתעת שם גודל לשימוש. ואותה לעשות את במקורה שאמו רצים לנקז את שמו של שוב, משתמש במילה `using`, על מנת לשלוט עתה ב-namespace.

```
using System;
```

```
namespace mySpace
{
    class MyClass
    {
    }
}
```

```
class MainClass
{
    public static int Main()
    {
        mySpace.MyClass myobj = new mySpace.MyClass();
        return 0;
    }
}
```

אם נזהה לבקש את אורת היכתבה, ונsoftmax במליה `using`:

```
using System;
using mySpace;

namespace mySpace
{
    public class MyClass
    {
        public static void doNothing()
        {
        }
    }
}
```

interface .2.61

ניר אדר לא מונה מילוקה אבסטראקטית, שבה לא מוגדרם משתנים, וכל הפקטיות
ביהון אבסטראקטיתו. מטרת ה-`interface` רונה לגדיר התנהגות טרויומת, והוא משמש בסיס למלוקות
המשמעות המתואמת לו. מילוקה בינה לבין `interface` אובייקטיבית של הפקטיות בה אבסטראקטית, הא
הבדל בין מילוקה לבין `interface` מוסף מילוקה להגדר מתחילה בהודת בלבד. סיבת
הסיבה לכך היא ששים שוו והולשת, הורשה של המנגנון והורשה של מושך. סיבת
הסיבה לתמונות בוהגה של מושך, לטעמה שפה C# תומכת בורה מהירה ייודה שפה.
מימוש, ובורשה מרבבה של מילוקה המתואמת.

הגדת מילוקה הרשומה נילדי מילוקה הרשומה `interface`.

```
interface MyInterface
{
    void Function1(int iValue);
    int Function2();
}
```

או מגדיר את הפונקציות בתוך ה-`interface` כל הפקטיות הן `public` וכולל מגדירות כ-
`virtual` באוט אוטומטי, ויכן אזן לירשות דבר פרט ללחומרה של פונקציה – שם-
הפרמטרים של הורשה ועליהם מוחרים.

`internal` להויה `public` או `internal` `Interface`
`public` במשמעו שיעיר יהה להשתמש ב-`internal` היה באותו `interface` היה באותו `assembly`.
שמועה שיתן להשתמש בו בכלל `internal`.

```
namespace Nested // a nested namespace
{
    public class ClassInNestedNamespace
    {
        public static void SayHello()
        {
            System.Console.WriteLine("Hello");
        }
    }
}

public class UnNestedClass
{
    public static void Main()
    {
        MyAlias.ClassInNestedNamespace.SayHello(); // using
        alias
    }
}
```

```
Cow David = new Cow();
```

```
printMessage(Dogi);
printMessage(Smil);
printMessage(David);
```

```
static void printMessage(Animal aminal)
{
    aminal.Makesound();
}
```

IEnumerable,IEnumerator .3.61

ושא רוא נשא שוב ב-C#.NET מוגלים בעורם לאחאר על מחריקות שלן
כמחלקות הממשות התהנות מסימניות, ומלהמת אנטומיה.
הוגם הרואה ענראה רוא מינוש של interfaces-ה'ן
Interfaces. IEnumerable<Animal> IEnumerator<Animal>
עליה אוטו-הפקה באמצעות שפט הינה `.foreach`

כל שהשתמש באילו און איזים לאחאר את הפעולה הינה לאש הטעינה:

```
using System.Collections;
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

interface Animal
{
    void Makesound();
}

class Dog : Animal
{
    public void Makesound()
    {
        Console.WriteLine("Woof");
    }
}

class Cat : Animal
{
    public void Makesound()
    {
        Console.WriteLine("Myaow");
    }
}

class Cow : Animal
{
    public void Makesound()
    {
        Console.WriteLine("Muuuu");
    }
}

public class MyClass
{
    public static void Main()
    {
        Dog Dogi = new Dog();
        Cat Shmil = new Cat();
    }
}
```

:Interface-Λגונט

תבוט בודקן הממשתאות בinterfaces-הן

:IEnumerator

```

using System;
using System.Collections;

class colors : IEnumerator, IEnumerable
{
    private string[] cols;
    private int iCurrent;

    public colors()
    {
        cols = new string[] { "Red", "Blue", "Green", "White" };
        iCurrent = -1;
    }

    public IEnumerator GetEnumerator()
    {
        return (IEnumerator)this;
    }

    public bool MoveNext()
    {
        if (iCurrent < cols.Length - 1)
        {
            ++iCurrent;
            return true;
        }
        return false;
    }

    public object Current
    {
        get
        {
            return cols[iCurrent];
        }
    }
}

```

```

public interface IEnumerator
{
    bool MoveNext();
    object Current { get; }
    void Reset();
}

```

Public Properties**Current**

Gets the current element in the collection.

Public Methods**MoveNext**

Advances the enumerator to the next element of the collection.

Reset

Sets the enumerator to its initial position, which is before the first element in the collection.

:IEnumerator

```

public interface IEnumerable
{
    IEnumerator GetEnumerator();
}

```

Public Methods

Returns an enumerator that can iterate through a collection.

GetEnumerator

UnderWarrior Project	http://www.underwar.co.il
96 תרגום	

CompareTo
Compares the current instance with
another object of the same type.

הטחצזה ש-ה מגדיר פונקציה המאפשרת השוואת בין האובייקטים הקיימים בתוכו.

לאובייקט המתקבלי בפעמיין, ומחריה ערך של שום שים יתיחס ל'יבג', 'שלא' ואפס.

אם הערך המוחזר קיר מאפיין, אך לאובייקט המכתר לו זו מופתעת. אם הערך גדול מ-אפס, או האובייקט המכתר, אם הערך הוא אפס, או האובייקט המכתר שלו לא מוגדר.

אם נכתוב מולולקה המוממשת של `Interface`, מכל ליהן אובייקט מוגדר שלו, אם ניצור מערך של אובייקטים מסוגו, מכל לשימוש בטוקין `Sort` ששייכת ל-`Interface`, או מרת למת'ן את המערך.

Interfaces המגדיר `Interface` .5.61

יון להגדר הגדר מכמה `interface`ים אחרים. במקורה כה, מוחליק ש-`shemesh` יעצור מה `interface`ים ש-`shemesh` שיחזור בנהם.

ב-`C++`, אם במליליק שנות מוקה און גודרים מוליקה חדשה ש-`shemesh` יעצור מה `interface`ים ש-`shemesh` מוגדרות באותו מקום. ב-`C#`, לעומת זאת, אם גודר `interface` אחד מושך מוליקה חדשה ש-`shemesh` לא מוגדר `interface`ים.

Comparable

`IComparable` .4.61

`IComparable` אח שבסה `Comparable` ו-`Interface` כל מוליקה ש-`shemesh` אותו יכול להראות מוגדרת.

כבר דוגמא לפתרון של `C#` ללבנית, נדרש שי שמי `Comparable` בטלית אותה חתימה, וניצור מוליקה שהשווה ש-`shemesh` משלים, וכן יגיד און מעלים בהונגשנות השמות.

```
interface Interface1
{
    void myFunction(int i);
}

interface Interface2
```

yntax להשתמש במשפטים `using` ו-`Dispose` תקראי באפוי

```
using System;
using class2;
using var1;
new var1 s1;
class2 var2 = new class2(0,...);
}

void interface : interface1, interface2
{
    void interface1.myFunction(int i)
    {
        // Function body goes here
    }

    void interface2.myFunction(int i)
    {
        // Function body goes here
    }
}
```

!Disposable שימוש במודול 6.6.1

כפי שראינו, כאשר אם כותב פונקציה הרוותת, הא אל בהרבה וקරאת ברגע שאנו כרך אוור אובייקט, והיא נעה להרקרה גם מאוחר יותר...
לעתה יש שורר בשורה של מ"ד נושא בין מ"ד נושא צורן.
על מנת לבעזע את שמותם בפונקציית `Dispose()` – המשמשת לשחרור משאבם מ"ד, שוד לפנינו.

אם ק"מ אובייקט רתום מוחליקה הנגרת ה-`interface`-ה, ברגע שהוכנית המ משתמשת אובייקט זה ראה שהוא שאר אונגה ציראה אותו שד, הוא קוריאת פונקציית `Dispose()`, אונגה אונגה אונגה את פועל ה-`finalize` ה-`Garbage Collector`, וכך יתוקן שנותנצע פעלת מותנה.
כך לטעמך כה, ויתן לזרע עלי בחרן הפונקציה `Dispose()` גומגת ליכך שרטוטה הרוותת א"ל תעשי.

פונקציה מוגדרת ש"י `GC.Collect()` היא `Garbage Collector` פונקציה מוגדרת ש"י `GC.SuppressFinalize(this)` פונקציה מוגדרת ש"י `Garbage Collector` לשחרר את כל האובייקטים שאין יותר-used.

Exceptions - תרגום שגיאות .71

מבחן בדיקות שגיאות

.71. מבוא

```

}
}

static int Main()
{
    try
    {
        Console.WriteLine(Div(10, 2));
        Console.WriteLine(Div(4, 0));
        Console.WriteLine(Div(8, 2));
    }
    catch (Exception e)
    {
        Console.WriteLine(e.Message);
    }
    return 0;
}

```

הפקאה של DIV מולקה שי מוסרים ווחיריה את התחאה. במודה והוורש תחילה .Exception.

- ב-0 היא שולחת שגיאה של .Exception.
- ב-בן אם קווים פעמים לפונקציה של exception.
- בעם השניה תשליך catch-הקל.
- בבעה, וכייל לא מ踽ם אל הרקאה העששית.
- ב-בנ"ט יעתום לשינויים ב-C#-ב' אומ' exception מכל סוג. ב-אנו יカリים לישרו רק אובייקט exception או exception ממהלקה זו. או אנו יוציאים אובייקט exception שאירט אוטו.
- שילוח exception נעשה בעורף המילה השמורה .throw.

על מנת לבדוק יייחופו שאՅו, אן מושתמש במודה הבא:

```

try
{
    catch (Exception e)
}

```

כל תוכיות מרוכבת עריכה מוגבז בדיקת שגיאות מוסים – סופקאות שנות (כון פוליאיות המקצת זיכרין), או פוליאיות מסוימת לפחות שעילית לרקייה (סיניות דוד לדיין לו קורא אם הערך לא מוגזם להזיהה או נגלה). כתם קורר מנוקש רטורט בוגת השאייה ומונבה ביחסו.

בפונקציה, שג� זו ריו מוביל בוחומת שגים – הרוא מסובן את הרקע, (השה להבחין בין הרקע לברך).

בדקה הינה גולן (מקיט מושקים בעקב עבר הקזאתות אצ'וון דאמטיות מוכבת), מסקן לשחרר את הירקון שמיין שער ש"י ש"עך לשיאיה.

וונגו טופזיות של ערך מוחר ימיל להיל תיתם עם ערך אלי. ייטו.

לפוך C++ ובשבורתה גם מוכם במודע exceptions ו-C# בשבורתה גם מוכם במודע exceptions.

היעון הראהנו, בדומה לפונקציה ב-0, הוא הרפהה ב-0 העד ליקשע בדיקת השגיאות. אן קטע הרקע שוד שאמן מוחם או שום בתום בילק-אי. בודה ומתוחה שגאייה, השאה אל מוגזם מהפונקציה כנור מוח על המלה .throw return אלא נל' מוגן exception-הן. הטענה שפוקיק את הרקע שאירט אוטו exception שאירט אוטו, התענית תפוקיק את פועלנה.

מוגן exception שאירט אוטו – אם ונעל בה, התענית תפוקיק את פועלנה.

מוגן:

```

using System;
class MainClass
{
    static double Div(double fNumber1, double fNumber2)
    {
        if (fNumber2 == 0)
            throw new Exception("Divide by zero error.");
        return fNumber1 / fNumber2;
    }
}

```

כאשר קוראות בפונקציה כשליש exception המהוינה שלה המשמעות היא**exception**.

try

{

)

catch

{

throw;

)

לעתים נרצה לאפשר פעולות כלשינו, ונו אם קרא ל-**finally**, או אין בליך כלום. אם מתרוצזיה הקואוטר קליף מעלה ר-**try** שקיים לה. אם אין בליך כהה, או יוצאים גם מתרוצזיה הקואוטר קליף מעלה. בנסיבותnde שם מוגאים קוד המטפל בשיאיה.

finally:

try

{

Console.WriteLine(Div(10, 2));
/Console.WriteLine(Div(4, 0));
Console.WriteLine(Div(8, 2));

} return 0;

} catch (Exception e)

{
Console.WriteLine(e.Message);

} finally

{
Console.WriteLine("The End...");
}

בדוחה **Exception** דוחה ל-**C++**, אנו מוגאים לשים מספר בלוקים של **catch** אשר אחד אחריו, ליתפסת. אם קובלות שайיה בבלוק **catch**, תוביל לעשרות אותן הפעולות, ותיל עלשות אותן באותות מושן. מקרים הבאות:

Exception

בדוחה **Exception** הוא מושן וכוחב. בחרה שהאפקט הוא מושן שונין.

בחרה שהאפקט הוא מושן וכוחב:

```
try
{
    ...
}
catch (Exception e)
{
    Console.WriteLine(e.Message);
}
finally
{
    ...
}
Console.WriteLine("The End...");
```

ניתן גם ליתפס את כל סוג ה-**try** רשות **finally**. פשרה זו תרכה **C++** **catch(...)** זהה לרשום (

```

    Console.WriteLine(
        "Exception caught here" + e.ToString() );
}

```

```

Console.WriteLine("Last Statement");
}

```

ApplicationException .2.71

יש שות מתקינות כדי סבר שיאות ב-CLR, האות היא נט „Net“ הושיטה ה-CLR, הרשונה מעלה מהלקות מודגמאות מחריבת שגרה בחרביה קיימות ונוסין תקשורות לאו שאר שרים קיימים, וושיטה שורה היא שיאת שיאות בתכנית של המשמש, לא שאל שביבה הראשית, ExecutionEngineException, ArgumentException ו StackOverflowException באפוי ApplicationException עטוף, ולגבי StackOverflowException יתור מחר אין זו נוגע לדריך שיאות מוגזם.

משל Exception ת.3.71

ונט איר ליאו לשיטים שיאות משל בעיות מודע “חוד ליטן שומר בהן. כי ליעשנה איגר מוחקה מהתהנת מתנו.)

לול דגם לתוכחה שיאיה:

```

using System;
class Myexception : Exception
{
    public Myexception(string str)
    {
        Console.WriteLine ("User Defined Exception");
    }
}

class MyClient
{
    public static void Main()
    {
        try
        {
            throw new Myexception ("dugma");
        }
        catch (Exception e)
    }
}

```

Operators Overloading .81

ח. 1.81. אופרטורים ביאריים

```

    }

    public Complex(double dr, double di)
    {
        r = dr;
        i = di;
    }

    public static Complex operator +(Complex num1, Complex num2)
    {
        Complex ret = new Complex(num1.r + num2.r, num1.i +
        num2.i);
        return ret;
    }
}

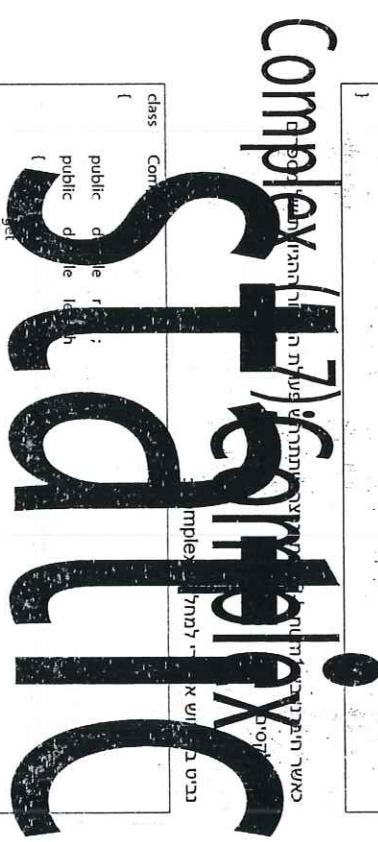
```

בוגרמו חפפו את האופרטור **+**
כפיות אופרטורים נמשת על ידי טווחאים סטטיות בלבד. הטעינה שכביכה
מקבלת שני מספרים קומפלקטיים, מתרבת אותם ומוחירה את התוצאה.

הפעת אופרטורים רינה מאפשרת להגדיר מוחש את הפעולות הבסיסיות על משתנות, כמו
- **+**, *****, *****, **/** שטויי אוטם על הוכנות שלן.
למשל, גווח שכתבת מחלקה המאפשרת מספרים קומפלקטיים, שוקרא לה:
ונזה לדאוג שונני ליכטב את הטעינה הבהאה:

```
class MainClass
{
    int Main()
    {
        Complex num1 = new
```

```
Complex(4.2, 0);
Complex num2 =
Complex(4.3, 0);
Console.WriteLine("num1 + num2 = " + num3.r + " + " + num3.i);
return 0;
}
```



האפרטור + רם אפרטור בפונק' רם אפרטור כמותו האפרטור * ו- / . לאנתרטם שם אופרטורים אובייקטיבים, +++ ו- ++, המקבלים +++ ו- ++, עלי' רם פוטעלים. לוגיאן, נס, שואיל אופרטור +++ הממש' של המדרת אופרטור +++ למלוקל, שואיל את הדרת המדרת אופרטור +++.

```
public static Complex operator +(Complex num)
{
    num.r++;
    return num;
}

public override bool Equals(object o)
{
    if (r == ((Complex)o).r && i == ((Complex)o).i)
        return true;
    return false;
}
```

```
public static Complex operator +(Complex num1, Complex num2)
{
    if (num1.r == num2.r && num1.i == num2.i) return true;
    return false;
}
```

```
public static bool operator==(Complex num1, Complex num2)
{
    if (num1.r != num2.r || num1.i != num2.i) return true;
    return false;
}

public override int GetHashCode()
{
    return this.GetHashCode();
}
```

שטע אופרטורים הבאים בוחות אם ווזה בה חישובו +=. להחישם את האופרטור += מוחיבת גם את המדרת =>. כמ' ק', המדרת > מוחיבת גם את המדרת =>, אליו המגרה מוחיבת גם את המדרת =>. מבר ואופרטור == אצטר לזרוק וט את האופרטור Equals עם זאת המדרת Equals מוחיבת גם את המדרת =>.

לגאנט מסכם:

```
class Complex
{
    public double r, i;
    public double length
    {
        get
        {
            return System.Math.Sqrt(r*r+i*i);
        }
    }

    public Complex(double dR, double dI)
    {
        r = dR;
        i = dI;
    }

    public static Complex operator +(Complex num1, Complex num2)
    {
        Complex ret =
            new Complex(num1.r + num2.r, num1.i + num2.i);
        return ret;
    }
}
```

3. פונקציות המהו

Indexer 2.81

C#-בן לאusa המהו בון טיפוסים בודדים שנות ידי casting ו-Indexer-ב שמתממש עלי מנה לחרוף אפורטו זה הגהה קדוקה היאן.

תוליל מאפשר המהו [2] אובייקטים שונים, ואילו אם המהו מתבצע באופן אוטומטי (implicit) או באופן מפרש (explicit) במאשעת אופטשו המהו.

הואר ההזדקקית להגהה הא ניכר.

```
public static implicit operator TargetType(SourceType s)
{
    get
    {
        return m_value[index];
    }
    set
    {
        m_value[index] = value;
    }
}
```

פונקציות המהו, בדומה לפונקציות הוחפות אופרטורים, חייבות ליתרונות הוחפות סטטיות.

```
public static string indexer(int index)
{
    if (index < 0 || index > strings.Length - 1)
        throw null;
    foreach(string s in strings)
        if (s[0] == start && s[s.Length - 1] == end)
            return s;
    return null;
}
```

אפשרות נספוח ראה הפיכת המהו לערך בלבד או לא-יאיה.

בלבד עלי השנתה קיד און set{} בלבד, אם אל מוחב בלהק set{} אל לא מוח.

לאריב ערכם בתגובה להק הרה ל�ראה בלבד.

אפשרות נספוח ראה העברת מסך מפרטרים לגודם, גנוב indexers, ומספר indexers.

לום אם המהו הראשונה שהטללה באתות מסוימות מסוימות באורה, אם אל מוחא רוחץ null:

```
public string this[char start, char end]
```

```
{
    get
    {
        foreach(string s in strings)
            if (s[0] == start && s[s.Length - 1] == end)
                return s;
        return null;
    }
}
```

כגונה שמהו הוחפות אל יקחן.

כבר אדר

```
public static int Main()
{
    PrintFunc prnFunc = new PrintFunc(Print1);
    prnFunc("Hello, World");
    prnFunc = new PrintFunc(Print2);
    prnFunc("Hello, World");
    return 0;
}
```

```
*****Hello, World*****
Hello, World---
```

הפל שיתקבל ביריה:

```
*****Hello, World*****
```

דגמא טופט:

```
delegate Return_Type DelegateName(parameters);
```

זה יוזר טופט דש שם שנקראה Shmackahaha מבער לשלוחו של שטחןDelegateName()

גומרים כי שטחן בסוגיהם, מושחה ער' זוג רט'ן. Return_Type ונת למגדר את delegate'ן ב簿ן מוחקה או מח לה.

```
using System;
delegate void MyDelegate();
delegate void MyDelegate();
```

```
public class MyClass
{
    public void InstanceMethod()
    {
        Console.WriteLine("A message from the instance method.");
    }
}
```

```
static public void StaticMethod()
{
    Console.WriteLine("A message from the static method.");
}
```

```
public class MainClass
{
    static public void Main()
    {
        MyClass p = new MyClass();
    }
}
```

UnderWarrior Project

<http://www.underwar.co.il>

טיפס וטיפים בAIR עם Delegates .91

Delegate .1.91

פוקאה ש' אובייקט. וכך זה מחליף שנות' יוצרים לחדב בינו מוצבעים אל הטעאות שוליה.

אזהה הורקחת ש' הרה כליה:

```
delegate Return_Type DelegateName(parameters);
```

זה יוזר טופט דש שם שנקראה Shmackahaha מבער לשלוחו של שטחןDelegateName()

גומרים כי שטחן בסוגיהם, מושחה ער' זוג רט'ן. Return_Type ונת למגדר את delegate'ן ב簿ן מוחקה או מח לה.

```
using System;
delegate void PrintFunc(string s);
class MainClass
{
    static void Print1(string s)
    {
        Console.WriteLine("*****{0}*****", s);
    }
    static void Print2(string s)
    {
        Console.WriteLine("----{0}----", s);
    }
}
```

UnderWarrior Project

113 טען

<http://www.underwar.co.il>

דעתא:

```

using System;
// Step 1. Declare a delegate with the signature of
// the encapsulated method
public delegate void MyDelegate(string input);

// Step 2. Define methods that match with the signature
// of delegate declaration
class MyClass1
{
    public void delegateMethod1(string input)
    {
        Console.WriteLine(
            "This is delegateMethod1 and the input to the
method is {0}", input);
    }
}

public void delegateMethod2(string input)
{
    Console.WriteLine(
        "This is delegateMethod2 and the input to the
method is {0}", input);
}

// Step 3. Create delegate object and putting in the methods
class MyClass2
{
    public MyDelegate createDelegate()
    {
        MyClass1 c2 = new MyClass1();
        MyDelegate d1 = new MyDelegate(c2.delegateMethod1);
        MyDelegate d2 = new MyDelegate(c2.delegateMethod2);
        MyDelegate d3 = d1 + d2;
        return d3;
    }
}

```

כארן יוזרים אנו יוצרים למשה אובייקט שמליחת המהלך .delegate

שנתה	מפעין / פונקציה
אם הפקיצה המתבצעת למולחה,	Target
ויהו שום המולחה אם הפקיצה הינה	
פונקציה סטטית, יותר NULL.	
מחזיר את שם הפונקציה שהפכה-	Method
מבעעליה.	
מחבר שת פונקציות כך שתה	Combine()
יעשי אותה.	
הפקיצה מודירה וטוקר של Delegates	GetInvocationList()
כלש נויסת מיצעת מבעעל פונקציית	
.Delegate הינה שטיפל	
מושעה מבעעל לפונקציה מהותה	Remove()
המוחק.	

2. סינפל באירועים

טיפר באירועים בהלונות רה אדר המשמשים החשיבות ביותר.
ב-[C#](#) הינה הדרישה הממליצה באירועים הרה `delegate`

ונמה בדוגמא:

```
using System;
namespace Events
{
    class Sink1
    {
        public int OnAdd(int x, int y)
        {
            Console.WriteLine("x+y={0}", x+y);
            Console.WriteLine("Input a number");
            return x+y;
        }
    }

    class Sink2
    {
        public int OnMul(int x, int y)
        {
            Console.WriteLine("x*y={0}", x*y);
            return x*y;
        }
    }

    public int OnAvarage(int x, int y)
    {
        Console.WriteLine("Avarage x,y={0}, {(x+y)/2}");
        return (x+y)/2;
    }
}
```

```
//Step 4. Call the encapsulated methods through the delegates
class MyClass3
{
    public void callDelegate(MyDelegate d, string input)
    {
        d(input);
    }
}

class Driver
{
    static void Main(string[] args)
    {
        MyClass2 c2 = new MyClass2();
        MyDelegate d = c2.createDelegate();
        MyClass3 c3 = new MyClass3();
        c3.callDelegate(d, "Calling the delegate");
    }
}
```

המכנית מופסיה למתן את הפונקציות הרשומות בברוח Event-לי. לאחר מכן נפעילה את ה-`event`.

לآخر תום הפעולה את ה-`event` לאיים אין בוך הערך.

```
class Test
{
    public delegate int Func(int x, int y);
    public static event Func event1;
    public static event Func event2;

    static void ShowAllFunctions(Func e)
    {
        Delegate[] arr=e.GetInvocationList();
        Console.WriteLine("methods in event");
        foreach(Delegate d in arr)
        {
            Console.WriteLine(d.Method.ToString());
        }
    }

    static void Main(string[] args)
    {
        Sink1 s1= new Sink1();
        Sink2 s2= new Sink2();
        event1+= new Func(s1.OnAdd);
        event1+= new Func(s2.OnFull);
        ShowAllFunctions(event1);
        int z=event1(5,6);
        Console.WriteLine("z={0}",z);
        event2+=new Func(s2.OnAvarage);
        event2(5,7);
        event1-= new Func(s1.OnAdd);
        event1(10,20);
    }
}
```

תבונית הגדרת שת מהדרית פותחים שיגיבוט .Events-לי שמתוליהן המדריך שמתוליהן .Sink2-ו Sink1 המולוקת הינה .OnAdd, OnMul OnAvarage הינה .event1, event2 :events מה צורת שנו events

- אתה מתקבץ לשיט קשי', אתה מושך בשין מספירים וכך אתה שומר את המספ"ר "בראי" (הדור הראשון).
- אין תבנה אליה ציר למשורר תאורה יותר, בוגדים אתה מתקבץ למשורר אונש לש וספס. (האור לשלוט לאחרים הוא מודע למתבונר, אתה מוגן לשן לך 10 מספרים שונים בזמנית, אתה מיע לנקודה קרייטית בה אתה מתחם לרשומות אתה שומר לך לוח פתק (הדור גות שני).

- כל של של מספר הפקתיות הולג גאר, וזה מחייבות לסתור את הרשון שלן.
- השלון של התמליל עד אפס קווק, אתה מוחלט לעשוט סדר בmorpheme ומוחמי לעבר על הפתקיות, מוחלי מוחן אתה חזק את אלה שפה שמשן את השאש אתה רעם ובשבוב הכתבות (הדור השלישי) לשורה יורה ממושברת.

2. האם ניהול היזכון לא מהבען בשביבלי?

מג'ה חיכו פועל מוחלה לפחות לא ח'יבם להאר אוון, אבל יש לשיס לבלה מה שפה ומשך אוות בסבבון פל'. ההמתה מלפני מעלה כלומה אפשרה לחוץ למורה, "ער' פרטס הקטעים", במקיר האור ע"ד' יש שאור בשיטתה מיל' מעלה עלי' הרפטים האדול'ם".
את שאלת שטאלת בהרבה אשוד תחומיים, האם זה נכון לדעת מה קורא מתנות למכוסה מהו ש"ל נוהג במכונית? מתנות לא באמת צורך לדאג אפ' פעילת סבבב הרצאה – ואישלי' באנס...?
כך אפשר לנהוג במכונית מבלי' לדעת איך פועל המנוע, אך אם באנס לחזור בגאות מראים מזאע נעל' המתה ש'ק אפל' על המכלנית, ואם נוחור לעלטנו און, אם באנס להסוק בפיקוח קצינות הדעת ופשתות און צור בהתעשרות בתהלה ני' היל' היל'ו. אך אם באנס לעסוק בפקוח אפל' אקלזיות גודחות בעלות באעיטים קרייטים עיל' "ליל'ו".

2.02 הדורות החדשניים

10 יחס ברא בין הדורות יגדיר כך: מספר האבקיטים בדור 0 היה 1, ברור 1 היה 1, פוט מהה 0 בראיה ייחד בדור כל מספר אליטות הענין, פינ' רמה 1 יתבע לאוון למלעה מה 30-ה מיליטות השנויות, פווי' רמה שתיים ליקוח זו צאן מה כתילות בchnint.

כל להו אפקטיב ביחס לתפקיד סביבת בות. מספה להבון "מה קורא", ומה עלייה לעשנות שר מהה בעיטים בגבורת האבקיטים בחדרות הנמהה. האבקיטים מושגים עלי' לה'ישאר כל משל זון ההפיצה של הוניגת בחרילון ולעומת שמי' שמי' מאר' קא, אך אם אבוקיט ח'זון אירק' יורה, מוגבהת בדקיה אם יש אופשרה לפרט מלה? עיטים רוחות יtier, ובכך מוסכים המשאים הדוחשים לבי'קה. הוניגת זו מושגת ני' היל' העירמה (בר' שמראים הדוחשים לשליטה דורות, דורות 2-1, 0, 1, 0) נתקן למחוש את הוניגאות עלי' ד' המאטופור האבאה:

פיך זה נלבב גול' שושן כב' גושר נ' נ' גור אדר.

סיפים לניהול זיכרון כמה .02

ניר אדר**דו"ג עבר טעינה.**

- דו"ג 2 עבד דוחסה (המ"מ הנטה מופיע בספרים מודך הדפסים שליכם, ווחופטם כל שאל ישאר רות, ובכל פוטו מופיע בטופ המחר).
- דו"ג 1 עבד שאסיא.
- דו"ג 1 עבד שאסיא.
- דו"ג 0 כוונן.
- דו"ג 0 עבר שאסיא.

נסירתה הששללה. אם לא אובייקט ב-HDD אין זה אמור שרבז לאקייטים אוטומטית – לא בכחיה, ורק אובייקט סלול לילר הרבה מאד מזו. אך גניזה גורם גיר קה' משבשים אובייקטים אחרים, המodus השומר בתאום מאותו לאובייקט מוג מחורחת, וכך, לעומת מקרים מסוימים לעין בחושך, אין הנה בעירמת האובייקטים הגדרה זו.

6.02 מסוף טיעון

בשבי מה להזכיר את התהילה והדוחות אם לא בשבי לילן או ת'ר"ד י"מ , אבל מה צאן לעשיות?

הנושא הזה יכול להיות מוחות הרבה יותר, אבל זה נראה הקרטון שנינו למסטר זה, מבוטה אני שיל קוויאי והתקדם מתקאים באם יתנו, בוך, למצען מקרים מסוימים לעין בחושך, אין הנה ספר העשוות מרכיזות.

4.02 מקטני זיכרין

גרימתה מזקקה תערין במקטעים, אשר גודלים תהי' בהגדלתם. אם הגדיר > gcsServer<, אשר גודלים יוגבזים, יוגבזים מוגזרים החיצין במקטעים שאל. 16MB מודול הינה, 64MB אורתה הינה, 32MB בערימת האבוקטם גודלים מוגזרים החיצין במקטעים שאל. רק אובייקטם בדרכו 2 בערימת האבוקטם הגדים יגולפם להשתוע נעל, פג מסוף מקטשען.

5.02 מה קורא בזוז איסוף צבל?

הילג האיסוף, כייל מ-ה-HLO.

- האובייקטים ב-HLO מוכרים: כ' אובייקט נערך, אם אין מעבים אליו הוא מושען.
- כמוך לאוונן.
- ה-HLO מושען: כ' אובייקט'ם יוסוגו משוחרר ממהריה. • ה-HLO לא שבע דוחסנה.
- דו"ג 2 עבד סוחין.

3.02 שערת ואובייקטים הגדרה.

סקל על המבנה שהאג'אג'ה ישרה גם עירית האובייקטים הנגלה (HLO), בה שמותה אובייקט ב-טוליג'ה-מ-000-000 בתרם, בערימה זו מוגאנ אסיק ביל פעלם שיט שורז במקטע שאל שאל. האה הסבר בדוחשך. אך האה'קיטים שבאה לא דוחשים גולים והגדרה שרת'ה בבחזעון.

נסירתה הששללה. אם לא אובייקט ב-HDD אין זה אמור שרבז לאקייטים אוטומטית – לא בכחיה, ורק אובייקט סלול לילר הרבה מאד מזו. אך גניזה גורם גיר קה' משבשים אובייקטים אחרים, המodus השומר בתאום מאותו לאובייקט מוג מחורחת, וכך, לעומת מקרים מסוימים לעין בחושך, אין הנה בעירמת האובייקטים הגדרה זו.

6.02 מסוף טיעון

בשבי מה להזכיר את התהילה והדוחות אם לא בשבי לילן או ת'ר"ד י"מ , אבל מה צאן לעשיות?

הנושא הזה יכול להיות מוחות הרבה יותר, אבל זה נראה הקרטון שנינו למסטר זה, מבוטה אני שיל קוויאי והתקדם מתקאים באם יתנו, בוך, למצען מקרים מסוימים לעין בחושך, אין הנה ספר העשוות מרכיזות.

4.02 מקטני זיכרין

גרימתה מזקקה תערין במקטעים, אשר גודלים תהי' בהגדלתם. אם הגדיר > gcsServer<, אשר גודלים יוגבזים, יוגבזים מוגזרים החיצין במקטעים שאל. 16MB מודול הינה, 64MB אורתה הינה, 32MB בערימת האבוקטם גודלים מוגזרים החיצין במקטעים שאל. רק אובייקטם בדרכו 2 בערימת האבוקטם הגדים יגולפם להשתוע נעל, פג מסוף מקטשען.

5.02 מה קורא בזוז איסוף צבל?

הילג האיסוף, כייל מ-ה-HLO.

- האובייקטים ב-HLO מוכרים: כ' אובייקט נערך, אם אין מעבים אליו הוא מושען.
- כמוך לאוונן.
- ה-HLO מושען: כ' אובייקט'ם יוסוגו משוחרר ממהריה. • ה-HLO לא שבע דוחסנה.
- דו"ג 2 עבד סוחין.

4.6.02 מינימום מגניטיזם הולסית

כאשר לאבירי יתקיימו טקסים רוחניים, יש לונגץ'ה תרואה תיאתורית כארשתם כבר איטם בין הוהי. ס. עד כה הילך בסדר. לאחר האבוקט שאל עיבור לדור הבהיר משפטו שהוא איטם מוקד הייסורי, משמע על באבוקט מושגתו, משיסטי לדוד 1 ודוד 2 ללבנה מזקקה.

卷之三

卷之三

באלקטריקה או כטבוק לא יתאפשרה. להעתום יש אורך בו בוחרה GC.Collect() -לה. מוגדרת גורם מילוי פונקציית GC.Collect()

תגלו אוניברסיטה קיימת לומדנית זו הוא מטעם הרפרא של איזון זה.

ויאן 1.12

לעתות אחד לoston המודול על שמו `File`. למחרת שונינו לoston כה גנום ורובה עלי מילויים.
וזה רק שטחה מהשפה הנדרשת לוחטנותה שהעובי דורי סטורנו אצל רובב קד
ויתמצענות.

ופoor וטעאים וטענים שלא זיהו בטעון זה שטחים אוניברסיטאיים.

- Windows Forms – כדי לעבוד אונטומת פירסום כר' פירסום כו, תיכון Windows Forms ופונקציית `GetClipboardText` – כתה שורה – כתה שורה אונטוקה ועכברם של SwapBuffers
- Windows Forms – הראת בחרמה בסיסית ברכבת Forms. הראת בחרמה בסיסית ברכבת Forms אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום כו, תיכון Windows Forms, Serialization – שמות הדעת אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום כו, תיכון Windows Forms.

- אונטוקה שיר הרה פירסום לאונטוקה פירסום כר' פירסום כו, תיכון Windows Forms, XML, Serialization – שמות הדעת אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום כו, תיכון Windows Forms.

- אונטוקה מתקשה כר' פירסום לאונטוקה פירסום כר' פירסום כו, תיכון Windows Forms, XML, Serialization – שמות הדעת אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום כו, תיכון Windows Forms.
- אונטוקה מתקשה כר' פירסום לאונטוקה פירסום כר' פירסום כו, תיכון Windows Forms, XML, Serialization – שמות הדעת אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום כו, תיכון Windows Forms.

כליה להקל מתקשה כר' פירסום לאונטוקה פירסום כר' פירסום כו, תיכון Windows Forms, XML, Serialization – שמות הדעת אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום כו, תיכון Windows Forms.

כליה להקל מתקשה כר' פירסום לאונטוקה פירסום כר' פירסום כו, תיכון Windows Forms, XML, Serialization – שמות הדעת אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום כו, תיכון Windows Forms.

כליה להקל מתקשה כר' פירסום לאונטוקה פירסום כר' פירסום כו, תיכון Windows Forms, XML, Serialization – שמות הדעת אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום כו, תיכון Windows Forms.

ויאן כו, תיכון Windows Forms, XML, Serialization – שמות הדעת אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום אונטוקה ועכברם כחומר ברכבת CLS-פירסום כר' פירסום כו, תיכון Windows Forms.

C# Dictionary

מונח	משמעות	מונח	משמעות
שפה	שפה	שפה	שפה
בינהים	עלית	בינהים	עלית
הסבר	הסבר	הסבר	הסבר
כדי להריץ	כדי להריץ	כדי להריץ	כדי להריץ
דוגמאות	דוגמאות	דוגמאות	דוגמאות
0/1	Assemble	C#	VB

Word Translation Meaning

מונח	משמעות	הסבר
מוגלה	מוגלה	מוגלה
קומפליזיה	שגיאת קומפליזיה	שגיאת קומפליזיה
שגיאת זמן ריצה	שגיאת זמן ריצה	שגיאת זמן ריצה
חריג	חריג	חריג
שגיאה לוגית	שגיאה לוגית	שגיאה לוגית

Framework	Framework
.Net framework runtime	.Net framework runtime
Managed-code	Managed-code
MS-IL	MS-IL
Exe	Exe
executable	executable
OOP	OOP
Object	Object
Oriented	Oriented
Programming	Programming
JIT	JIT
Just-in-time	Just-in-time
Exception	Exception
CLR Common Language Runtime	CLR Common Language Runtime
Class-Library	Class-Library
Visual studio	Visual studio
Encapsulation	Encapsulation
Inheritance	Inheritance
Polymorphism	Polymorphism
Re-Use	Re-Use
Class	Class
Object	Object
Instance	Instance
Value-Type	Value-Type
Reference-Type	Reference-Type
RAM	RAM
Solution explorer	Solution explorer
modifier	modifier
Convention	Convention
Case-sensitive	Case-sensitive
Const – Constant	Const – Constant
Read-only	Read-only
Property	Property
Enumeration	Enumeration
Current	Current

W a-fch

אנו

קורס קיז - #C

תכנית לימודים כללית

תוכן

1	קורס קיז - # C
1	תכנית לימודים כללית
2	גילון בקרה
3	הוראות כלליות לתרגילים
4	שיעור # 1 – DOT.NET,OOP,Class
4	ראשי פרקים
6	תרגיל # 1 – מלבן ותאריך
8	תרגיל # 2 – תיק ותמונה
9	שיעור # 2 – SV, הכלה, Static
9	ראשי פרקים
11	שיעור # 3 – מערכיים איסופים
11	ראשי פרקים
11	תרגיל # 1 – Family
13	תרגיל # 2 – מערכיים ורשימות
14	תרגיל # 3 – תלמידות בקורסים
15	שיעור # 3 – הורשה
15	ראשי פרקים:
15	תרגיל - "הagina של" - Basic OOP in C#
17	שיעור # 4 – פולימורפיזם
17	ראשי פרקים:
18	שיעור # 6 – virtual & abstract
18	ר' צורתיות – Polymorphism – עבודה
21	תרגיל צורות
22	שיעור # 7 – Interfaces
22	ראשי פרקים:
23	שיעור # 8 – תוספות
23	ראשי פרקים
25	משתק זיכרון
28	תרגילי חזרה ל מבחן
32	חלק א – מחלקות
33	חלק ב' הורשות וממשקים
40	כל הבחנים

גילון בקרה

#	נקודות	שם	מספר הביקוד בבדיקה	נקודות בתרגול	ס"ב	נקודות בבדיקה	ס"כ
1 כ"ד בتمוז כ"ז ג'ת'	2	נירן Family 친구들	—	0-חיסור 1-חלקית 2-מלאה	0-לא בוצע 3-אינו רצ- רצ	מעמוד עד עמוד	0-חיסור 1- אייחור 2- מלאה
2 כ"א בتمוז כ"ג ג'ת'	2	—	8 ג'ת'	—	—	—	—
3 כ"ד בتمוז כ"ה ג'ת'	2	—	—	—	—	—	—
4 כ"ה בتمוז כ"ד ג'ת'	2	—	22 ג'ת'	—	—	—	—
5 כ"ג בتمוז כ"ט ג'ת'	2	נירן nh3	10,5 ג'ת'	—	—	—	—
6 כ"ז בتمוז כ"ה ג'ת'	—	—	—	—	—	—	—
7 כ"ח בتمוז כ"ו ג'ת'	—	—	—	—	—	—	—
חזרה							
מבחן							

שם:

חתימות:

הוראות כלויות לתרגילים

שים לב בכל המחלקות להקפיד על כתיבת מקובלות וברורה:

- שם משמעוטי למחלקה, בלשון יחיד, יתחיל תמיד באות אנגלית גדולה.
- שמות השדות (DataMembers) משמעוטיים, באותיות קטנות. (שמות עצם, מערכיים /
אוסףים – ברבים)
- שמות פונקציות משמעוטיים (שם פעולה + שם עצם), מתחילה באות גדולה, ומילה חדשה
באות גדולה.
- שמות שדות או פונקציות המורכבים ממספר מילים, כל מילה ברצף תתחילה באות גדולה
לדוג' השדה: bornYear או הפונקציה: ()()
- המנוע מכפל קוד. כל שורה שנכתבת יותר מפעם אחת אמורה להיות במתודה.
- חלוקת למетодות - מתודה אמרה לבצע דבר אחד ורק אחד. אם התקשית בניתנת שם
למתודה - נראה שעלייך לפצלה.
- ב main צריך להיות שימוש בכל המחלקות, משתנהן וממדודותיהן. כל מתודה שנכתבה
במחלקה - ולא הופעלה ישירות או בעקביפין דרך ה Main - למשה - לא נבדקה
תקינותה.

שיעור #1 - DOT.NET,OOP,Class

[תכון מערך שיעורים לקורס הבא אחרי קורס C# בסיסי]

ראשי פרקי

★ הקדמה: סביבת Net..

• JIT

• CLR

• Msil

• שפות בנה,

Class library

- OOP ★

• שיטת חסיבה בתכנות, כמו בעולם...

• עקרונות

Encapsulation - מחלקה ★

• הגדרה: תבנית לצירת אובייקטים, טיפול נתונים משוכל

• חברי מחלקה

Hiding – Encapsulation, הסורה , הכמה

• מציננו גישה

Access modifiers ★

• public

• private

• defaults

Instance - מופע ★

• שימוש המחלקה,

• משתנה מטיפוס המחלקה

Ref type

– Data Members ★

• Val – פשוטים

• Struct – תאריך

• ref. – מטיפוס מחלקה – Address. מושג ה Helvetica

enum

(תץ.) Readonly

const

Methods - שיטות ★

• פעולות שהאובייקט מבצע

• העממת שיטות - בירוי סל גלאן ווילס ציון ויליאם

void void void void

ref, out, params, optional

מיילת המפתח this, השימוש בה, נשלחת כפרמטר בסתר לפונק'

Constructor ★

בנאי בירית מחדל

העומסת בנאים

השלבים ביצירת אובייקט

Dtor - תפנית★

- Types ★

Value type & ref type

default values

- Stack & Heap

הנחייה מוקדמת ומיידית
הנחייה מוקדמת ומיידית
הנחייה מוקדמת ומיידית
הנחייה מוקדמת ומיידית

reference ref ref ref ref ref ref ref

ההנחיות הינה ref ref ref ref ref ref ref ref
ההנחיות הינה ref ref ref ref ref ref ref ref
ההנחיות הינה ref ref ref ref ref ref ref ref

- פונק' פונק' פונק' פונק' פונק' פונק' פונק' פונק'
פונק' פונק' פונק' פונק' פונק' פונק' פונק' פונק'
פונק' פונק' פונק' פונק' פונק' פונק' פונק' פונק'

Console.WriteLine("Hello World")
static void Main(string[] args)
{
 Console.WriteLine("Hello World");
}

read

תרגיל #1 – מלבן ותאריך

1. המחלקה מלבן

מאפיינים: אורך, רוחב

פונקציות: הדפסת ערכי המלבן,

properties לשני המאפיינים

чисוב שטח(1) – מחשבת ומחזירה את שטח המלבן.

чисוב שטח(2) – מקבלת מידות (מטר, קילומטר) מחשבת ומחזירה השטח במידה
המבוקשת.

чисוב היקף – שמי פונקציות כמו חישוב שטח.

ציור המלבן(1) – מצירת מלבן כוכיות ע"פ ערכי המלבן.

ציור מלבן(2) – מקבלתתו ומציירת מלבן מותzáה.

הוסף פונקציית `ToString` והציגו את השימוש במחלקה בעזרת ה `Console`.

2. המחלקה תאריך

מאפיינים: יום, חודש, שנה,

מספר התאריכים שנבנו (`private static`), תאריך בניית המחלקה (`public static`)

`enum eMonth`: טיפוס נתונים שמכיל את שמות החודשים.

`eDatePart`: טיפוס נתונים שמכיל את חלקו התאריך: יום, חודש, שנה.

פונקציות:

Properties לשולשות המאפיינים, כולל בדיקות תקינות.

בנאי(1) – מתחול תאריך בערכים התחלתיים כלשהם

בנאי(2) – מקבל שלושה ערכים (יום, חודש, שנה) ומתחול התאריך בהתאם.

בנאי(3) – מקבל שני ערכים (יום וחודש) ומתחול התאריך בשנת 2006.

בנאי(4) – מקבל ערך מסווג תאריך ומתחול התאריך החדש בהתאם.

`Print` – ממחזירה את ערכי התאריך בהתאם: `y/m/d`

`DateAdd(1)` – מוסיף ערך לתאריך

פרמטרים: סוג ההוספה (יום, חודש או שנה)(העוזר ב `eDatePart`), ערך להוספה.

`DateAdd(1)` – מוסיף 1 לתאריך.

פרמטר: סוג ההוספה (ככ"ל)

מקבלת מערך תאריכים ומחזירה המוקדם מביניהם.

פרמטרים: מערך מסוג `params` של X תאריכים.

מחזירה: התאריך המוקדם.

– השוואת תאריכים, פונקציה סטטית.

פרמטרים: שני תאריכים, שלושה פרמטרים מסוג `ano` להחזרת הפרש (יום, חודש

(שנה)

מחזירה ערך בוליаниי קבוע מיהו התאריך המוקדם.

.`Console` – הוסיף פונקציית `main` והציג את השימוש בחלוקת בעזרת

תרגיל #2 – תיק ותמונה

1. המחלקה: תיק

משתני המחלקה: משקל, צבע, דגם, גיל יעד, נפח, מספר תאים, שנת יצור, האם יש אחריות, מחיר.
פונקציות: הדפסת נתונים תיק.

чисוב האם משקל התיק תואם את גיל היעד. (モותר לשאת תיק שמשקלו עד 10% מהגיל).
 חישוב האם משקל התיק תואם את משקלו. (מקבלת: משקל, מותר לשאת תיק שמשקלו עד 2% מהמשקל).
 פונקציית רכישת תיק (מקבלת סכום בש"ח, ומחזירה את העודף).

properties לכל משתני המחלקה.

הօספי פונקציות חיון והציגו את השימוש במחלקה בעזרת ה `Console`. עליך להגדיר תיק גן, תיק ב'ג', 2 תיקי סמיור זיהום בצבאים שונים, מזוודה מחשב נייד, מזוודה טיסות, מזוודה נשיאה.

2. המחלקה: תמונה

משתנים: גובה, רוחב, גושא, סוג תמונה(אמנות/ צילום/ ציור/ אלבום), סוג צבע (פסטל פנדיה שמן אקריליק מים גואש) מספר התמונות שבמארגר (`private static`), תאריך בניית המחלקה (`public static`)

`eColorType` טיפוס נתונים שמכיל את סוגי הצבאים

`ePictureType` טיפוס נתונים שמכיל את סוגי התמונות

פונקציות:

Properties למשתני המחלקה, כולל בדיקות תקינות.

בנאי(1) – מארח תאריך בערכים הותחלתיים כלשהם

בנאי(2) – מקבל שלושה ערכים.

בנאי(3) – מקבל שני ערכים ומתחילה ערך דיפולטיבי לשלייש.

בנאי(4) – מקבל תמונה ומארח תמונה תואמת- העתק.

– ממחזירה מחוזות של הערכים.

בהצלחה!!!

שיעור # 2 - VS, הילה, Static

ראשי פרקיים

Visual studio ★

- ↳ פתרון ופרויקטים
- ↳ מבנה התיקיות
- ↳ *.csproj, *.sln
- ↳ הוספה קבצים לפרויקט
- ↳ הצגת והסרת קבצים
- ↳ תבניות פרויקטים
- ↳ פתיחת פרויקט ב net
- ↳ קובץ ומחלקה
- ↳ הפונקציה main

Properties ★

- ↳ פרטיים המתאימים את האובייקט
- ↳ Get-set או אחד מהם
- ↳ מאפיינים אוטומטיים
- ↳ קיצורי דרך
- ↳ בקרת נתונים - בדיקות תקינות
- ↳ מאפיין מחושב

Console ★

Strings ★

string builder ↳

struct ★

VT ↳

Constructor ↳

ENUMS - רשימות ★

Val-type, struct ↳

Flags ↳

המרות ↳

קונקרטיים – גורם קבוצה של מודולים

Static ★

משמעות static – מוגדר כGLOBAL – הינה גלובלי

static data member

Static class

Static property

– פועל בפניה הראשונה למחלקה, לאתחול משתנים סטטיים.

Operators overloading ★

Unary

binary

Indexer

Names conventions ★

ניהול מחלקות ★

Namespaces, הגדרה, יבוא

namespaces

תיקיות בפרויקט

namespace – כינוייםAliases

מחלקות מקוונות – nested classes

שיעור #3 - מערכים אוסףים

ראשי פרקים

★ מערכים

- ₪ מטיפוס ערך, מטיפוס מחלקה, מטיפוס רשימה
- ₪ פונקציות של מערך
 - ₪ גודל המערך
 - ₪ אתחול מערך ותחול האברים
 - ₪ סריקת מערך בולאה גילה וforeach
 - ₪ המחלקה array, המאפיין length
 - ₪ מערכים רב מימדים – הגדלה ותחול
 - ₪ מערכים של מערכים – jugged arrays (להראות)

מצגת מערכים (Command-Line Argument) אוסףים ★

- ₪ יתרונות
- ₪ ג'נרים ושאים
- List
- Stack
- Heap
- Dictionary

תרגיל # 1# - Family

כתבי מחלוקת המציגת משפחה: Class Family :

DataMembers ★ - תכונות:

- קוד משפחה
- שם המשפחה
- שנת הנישואין
- מס' ילדים
- אזרחות (מחוזות)
- מערך הכנסות לפי חודשים (גודל 12)

int[] salaries=new int[12...]

שורות ההגדירה:

- <List<int> agesChildrens=new List<int>()

11

* פעולות (פונקציות)

- בניית ריק ובנאי שמקבל ערכים (קוד, שם, שנת נישואין, אזרחות, מס' ילדים מאופי במבנה)
- פון get -set לכל תכונה. (ללא האוספים)
 - קוד המשפחה חיב להיות מס' בן 3 ספרות, אם לא-הודי עלי שגיאת לתוכנה של מס' ילדים אל אפשרות השמה (set). תכונה זו תעדכן בהוספת גילאי ילדים.
- פון (void) UpdateSalaryInMonth(int month,int salary) מקבלת חודש וסכום ומעדכנת את המשכורת בחודש זה.
- פון () ReturnSalaryToMonth() מקבלת מס' חודש ומחזירה את המשכורת בחודש זה.
- פון () AvgSalary() המחשבת ממוצע הכנסה לחודש.
- פון () AvgSalaryPerPerson() המחשבת ממוצע הכנסה לחודש לנפש.
- פון () YearsMarriage() המחשבת ומחזירה את מס' שנות הנישואין
- פון () PrintSalaries() המדפסת פרטי המשכורות: מס' חודש וו המשכורת
- פון () PrintChildren() המדפסת את מס' הילדים ופירוט גילאים.
- פון () Print() המדפסת פרטי משפחה: קוד, שם, אזרחות, מס' שנות נישואין (זמן פונקציה), מס' נפשות, הכנסה חודשית ממוצעת. פרטי משכורות (זמן פונקציה) ופרטי הילדים (זמן פונקציה)
- הוסיף קיום תוחמים לפני ואחרי ההדפסה: ("-----")
Console.WriteLine("-----");
- פון (void) AddAge(int age) להוספה גיל של ילד (הfonקציה מקבל גיל ותעדכן אותו ברשימה וכן תוסיף למס' הילדים)
- פון (bool) CheckAge(int age) מקבלת גיל (מספר) ומחזירה אם יש במשפחה ילד בגיל זה.
- פון () Count18Over() המחזירה כמה ילדים מעל גיל 18 במשפחה.
- פון (int) countUnderSalary(int salary) מקבלת מס' החודשים בהם הינה הכנסה מתחת סכום זה.

תרגיל #2 - מערכים ורשימות

תרגיל - Collections

בני את המחלקות הבאות:

1. מחלקת ציון

מכילה את הנתונים מקצוע (מתוך כמה של מקצועות) וציון באחוזים (0 – 100), הגדרי משתנים Properties מתאימים.

2. תלמידה

מכילה את הנתונים שם פרטי ומשפחה, כיתה ואוסף של ציונים כמו"כ מכילה את המתודות הבאות:

1. הדפסה – מדפסה את פרטי התלמידה ואת כל הציונים שלה.
 2. הוספת ציון – המתודה מקבלת ציון, בודקת אם כבר קיים ציון זה, אם כן מחזירה False אחרת מוסיף ומחזירה True.
 3. מחיקת הציונים ה"לא עוברים". לתלמידות עד כיתה ו' ציון לא עבר הוא מתחת 80, אחרת ציון לא עבר הוא מתחת 60.
 4. מין ציונים מהצין הגבוה ביותר לנמוך.
 5. מתודה שמחזירה את 3 הציונים הגבוהים ביותר (השתמשי במיון וב GetRange)
 6. מתודה שמקבלת אובייקט ציון ומחזירה היכן הוא מדורג בראשימת הציונים מלמעלה ומלמטה (כמה גבוהים ממנו וכמה נמוכים ממנו – השתמשי ב Out).
3. בפונקציה Main הגדרי מספר תלמידות אטחלי אותן בנתונים והפעיל עלייהן את המתודות השונות.
-

תרגיל #3 – תלמידות בקורסים

בסמינר למדעי המחשב לומדות עד 40 תלמידות. בסמינר מוצעים 20 קורסים שונים.

תלמידה מקבלת תעודה סיום רגילה כאשר עברה 10 מתוך הקורסים.

תלמידה מקבלת תעודה סיום בהצטיינות כאשר עברה 10 מתוך הקורסים, בממוצע הגבוה מ-80%.

תלמידה מקבלת תעודה סיום בהצטיינות יתרה כעבור 10 מתוך הקורסים, בממוצע הגבוה מ-95%.

יש לנහל את הלימודים באמצעות המחלקות הבאות:

מחלקה קורס

משתני המחלקה: קוד קורס (מספר בן 4 ספרות), שם הקורס, ציון עובי (בין 55 ל-75)

מחלקות ציון בקורס

משתני המחלקה: קורס, ציון (בין 40 ל-100)

מחלקה תלמידה

משתני המחלקה: ג.ז. (readonly), מספר תלמידה (בין 100 ל-1000), תאריך לידיה, מספר טלפון,

מערך של ציונים בקורסים, סטוסו (לומדת, סיימה, סימה בהצטיינות, סימה בהצטיינות יתרה)

מחלקה סמינר

משתני המחלקה: שם סמינר, כתובות, טלפון, מערך של 40 תלמידות

בכל אחת מהמחלקות הגדיר properties כנדרש.

את מערך 20 הקורסים ניתן למלא בחומר (במה שנסיר דרך מתאימה יותר)

ה邏輯ית נאפשרה:

- * הוספה תלמידה
- * עדכון ציון בקורס לתלמידה מסויימת, או לכל התלמידות. (יש לשים לב לטפל בסטוסו התלמידה בהתאם)
- * האגת הסטוס של תלמידה מסויימת, או של כל התלמידות.

שיעור #3 - הורשה

ראשי פרקים:

- ★ משמעות ההורשה. דוגמאות: עובדים, צורות
- ★ אופן הכתיבה
- ★ הרשות protected – מוכר לעצמו ולירושים
- ★ זימון בנאי של האב, זימון פונקציה של האב
- ★ סדר בניית אובייקט ירוש
- ★ הדגמת בניית אובייקטים
- ★ Sealed – מחלוקת שלא ניתן לרשות,

תרגיל - "הגינה שלי" - Basic OOP in C#

מחלקה צמח

מוש' מזוהה (readonly), שם, עונת גידול (שנתוי, חורף, אביב, סתו, קיץ) [enum flag], כמות
השקייה דרישה, תאריך שתילה, דישון (כן/לא)

פונקציות: גיל הצמח

מחלקה עצ: - ירוש מצמח

סוג (סרק, פרי) [enum], סוג עלים (משון, רחב וכו') [enum].

מחלקה פרח: ירוש מצמח צבע [enum]

מחלקה פרח חממה – ירוש מפרח

שם חממה, מספר שורה, מספר עץ, תדיות השגחה (enum – פעם בשעה, פעם ביום, פעם
שבוע)

א. בסיס

1. בני את המחלקות, בני את enum הדרושים,
2. הגדרי DataMembers,
3. הוסיף Properties, כתבי בהם בדיקות תקינות לערבים (כמות השקייה, גיל, תאריך
וכדומה)
4. בני שני Constructors לכל מחלוקת
5. כתבי פונקציית ~~details~~ details לכל מחלוקת (דרס במקורה הצורף)
6. הוסיף main, השתמשי במחלקות שבניית

ב. תוספות

1. שם הגינה הוא "הגינה המוחשבת". הגדרי `const data member` שיכיל מידע זה.

2. כתבי פונקציה המקבלת כפרמטר מספר N ומגדילה את גילו של העץ ב-`N` שנים. כתבי פונקציה זו פונקציה שגדילה את גילו של העץ בשנה אחת. העמיסו לפונקציה זו פונקציה שגדילה את גילו של העץ בשנה אחת. ~~לפונקציה זו ישנו מטרית נזקן~~

3. ניתן לשתול פרח על יד פרח אחר אם הם גדלים באותה עונה וצוקים אותן אותה השקיה. כתבי פונקציה במחלקה פרח שתתקבל כפרמטר פרח ותבדוק האם ניתן לשתול הפרח ע"י הפרח שנשלח.

4. כתבי ווריאציה נוספת לפונקציה הנ"ל שמקבלת שני פרחים ואין צורך ליצור מופיע בשביבה.

5. כתבי פונקציה למחיקת עץ מהזרה ערך בוליאני האם מותר להטוט ממנו (ערימה כן/לא – ללא דיקונים הלכתיים של חודשיים מס' 3 שנים)

6. ניתן לשתול עד 50 עצים. הוסיפו `static data member` שיעקוב אחר מס' העצים. הודיעו והודיעו מתאימה כאשר עוברים את המבנה.

7. כמות ההשקייה המותרת לכל העצים בגינה היא עד 500 ס"מ. כתבי `static function` שתתקבל את העצים (`params`) ותבדוק את מצב ההשקייה. הפונקציה תחזיר ערך בוליאני הקבע עם הגינה עומדת בקריטריונים הדרישים וכן (ב-`out` parameter) את כמות ההשקייה שהושבבה.

8. בנטיעת העץ והפרת הראשוני הדפיו ברכות לבני הגינה. (`static ctor`)

9. פרח חממה יכול לגודל ע"י פרח אחר אם מתקיימים כל התנאים הידועים וכן שביהם פרחי חממה וזקנים לאוותה השגחה. כתבי פונקציה דורשת שתבדוק זאת.

ג. **מחלקות גינה**

1. המחלקה תכיל מערך מצביים בגודל 100 מסוג _____, ומונגה _____ צמחים.

2. מוגדרת הוספה – המוגדרת מקבלת צמח ומוגיפה למערך.

3. מוגדרת הסרתה – מקבלת צמח ומסריזה.

4. מוגדרת הדפסה – מדפסה את נתונים כל הצמחים, אם מדובר בעץ עורלה יש להדפיס אזהרה באנוף.

5. מוגדרת ספירה – המוגדרת מקבלת את סוג הצמח הרצוי (פרח, פרח חממה, עץ וכו'), ומחייבת כמה מופעים קיימים ממון בגינה.

שיעור #4 - פולימורפיזם

ראשי פרקיים:

plant p=new Tree();
Free f=new Leaf();
f.setLeafValue("green");
System.out.println(f);
↳ מנגזרת לבסיס - גזירה implicit
↳ מנגזרת לבסיס - גזירה explicit
↳ מבסיס לנגזרת - גזירה explicit
↳ מבסיס לנגזרת - גזירה implicit

Upcasting, downcasting - IS - AS *

Tree t=new Tree();
Plant p=t

Tree t1=(Tree)p;

↳ המילה is - "האם מנהל הוא עובד"

↳ המילה as - "התיחס לעובד כמנהל"

* פולימורפיזם - אפשרות להפעיל מספר התנהגויות שונות באמצעות שם פונקציה זהה.

overriding | Overloading *

↳ overriding - דרישת פונקציות האב. דוגמא בprint וbySalary.

* המחלקה Object - כולם יורשים ממנה,

↳ פונקציות המחלקה Object

↳ דרישת הפונקציה ToString.

* העלמת פונקציות האב - New - בכתיבת מילת מפתח זו בכותרת הפונקציה במחלקה

היורשת הוא מעליימה את כל הפונקציות באותו שם ממחלקת האב, כולל כל הפעולות.

* פונקציה שלא ניתן לדרכו, פונקציה שחייבים לדרכו.

, boxing and unboxing *

* מהו virtual

Virtual property ↳

שימוש בפולימורפיזם - מערכת ממחלקת האב, הפניה לפונקציות ממחלקת האב והבנים

↳ abstract - פונקציה מופשטת

* מחלקה מופשטת - abstract

שיעור #6 - virtual & abstract

רַב צוֹרְתִיּוֹת – Polymorphism - עֲבוֹדָה

נתונות המחלקות A,B,C,D,E ו-Bן הפונקציות Func ו-DoSomething, כמפורט לעמם. בפונקציית ה-Main מוצאים אובייקטים מהמחלקות אלו ומתיבצעת קראיה לפונקציות שלתן.

רשמי לצד כל שורת קוד אם היא חוקית או לא, ובאם היא חוקית איזו פונקציה היא מפעילה.

```
abstract class A
1.   {
2.       public abstract void DoSomething();
3.   }
4.
5.   class B : A
6.   {
7.       public override void DoSomething()
8.       {
9.       }
10.
11.      public virtual void Func()
12.      {
13.      }
14.  }
15.
16.  class C : B
17.  {
18.      public new virtual void DoSomething()
19.      {
20.      }
21.  }
22.
23.  class D : C
24.  {
25.      public override void DoSomething()
26.      {
27.      }
28.
29.      public override void Func()
30.      {
31.      }
32.  }
33.
34.
35.
36.  class E : B
37.  {
38.  }
39.
40. //-----
41.
42. static void Main()
43. {
44.     A a1 = new A();
45.     a1.DoSomething();
46.     a1.Func();
47.
48.     A a2 = new B();
49.     a2.DoSomething();
50.     a2.Func();
51. }
```

```
52.     A a3 = new C();
53.     a3.DoSomething();
54.     a3.Func();
55.
56.     A a4 = new D();
57.     a4.DoSomething();
58.     a4.Func();
59.
60.     B b1 = new E();
61.     b1.DoSomething();
62.     b1.Func();
63.
64.     B b2 = new D();
65.     b2.DoSomething();
66.     b2.Func();
67.
68.     C c1 = new D();
69.     c1.DoSomething();
70.     c1.Func();
71.
72.     C c2 = new E();
73.     c2.DoSomething();
74.     c2.Func();
75.
76.     D d1 = new B();
77.     d1.DoSomething();
78.     d1.Func();
79.
80. }
81.
82.
```

הודשות שגאה – שיעור 1

Use of unassigned local variable 'p2' .1

השגיאה מופיעה בעקבות שימוש בmenoاع שעדיין לא הוקצתה לו זיכרון

הקוד כתוב הוא p1.name=dsafsd

The name 'dsafsd' does not exist in the current context
השגיאה המופיעות היא בשם שם
משמעותו: שם זה אינו קיים בהקשר הנוכחי (יש לשים את התווים במיראות או להציב שם
משתנה קיימת)

p1.address.city="Jerusalem" .3

.Object reference not set to an instance of an object
השגיאה המופיעות היא address
הסבר :ovic new, לא הוקצנו זיכרון לאובייקט address
תרגום מאנגלית

Object reference – התייחס לאובייקט, המצביע, המשתנה מסווג האובייקט
not set – לא הצביע, לא הוקצתה
instance - למוגע

הודשות שגאה ואזהרות – שיעור 4

1. אזהרה – לא שגיאת קומפליציה המתרחשת בעקבות יצירת פונקציה בשם זהה לפונקציה הקיימת
Lesson4_Inheritance.clsEmployee.age() override
במחלקה הבסיס לא שימוש בvirtual hides inherited member 'Lesson4_Inheritance.clsPerson.age()' . Use the new keyword if
hiding was intended.

תרגום חופשי – הפונקציה מסתירה את הפונקציה ממחלקה הבסיס אם הוכוונה לבר השרטוש
במילוי המפתח new

בז"כ מומלץ לא לעשות זאת כי זה מקלקל את הפלט מורפיזם
cannot derive from sealed type 'Lesson4_Inheritance.Class1'
לא ניתן לרשף ממחלקה שהוגדרה כ sealed
sealed - לרשף derive

תרגיל צורות

תיאור:

קבוצות המחלקות שתבנוי תתאor צורות שונות: מלבן, עיגול, טיבח, כדור וגליל. כל צורה תתואר בצעע ובממדים המתאימים לה (אורך, רוחב, רדיוס וכו'). כל צורה נאשף לחשב היקף, שטח, נפח, להדפיס את ערכיה ועוד. כמו כן ניתן לבצע פעולות חשבוניות על הצורות על ההגדרות דלעיל.

בנית המחלקות:

א. לכל הצורות ניתן להגדיר צבע וכן את הפעולות: ציור, הדפסה (toString | print), חישוב היקף ושטח. כתבי class abstract class "צורה" שתעננה על הדרישות. המאפיין – צבע, הfonקציות וירטואליות.

ב. המחלקות מלבן ועיגול, משלש הן צורות. כתבי מחלקות אלו כך שירשו את היכולות של "צורה". הוסיף את המאפיינים הדרושים לכל אחת (אורך ורוחב למלבן ורדיוס לעיגול). ממשי את כל הfonקציות. (fonקציית ציור של עיגול ותדפיס משפט בנוסחה: "אני מציר עיגול בממדים...").

ג. לכל הצורות התלת מימדיות יש פונקציה חישוב נפח ושטח פנים. בני interface "תלת מימד" שיתאר זאת.

ד. טיבח היא מלבן מיוחד מטיפוס תלת מימד. המחלקה "טיבח" תירש את "מלבן" ותתמש את "תלת מימד". הוסיף את המאפיין גובה, דרש את פונקציות המלבן, ממשי את פונקציות ה"תלת מימד".

ה. גליל וכדור הם עיגולים מיוחדים מטיפוס "תלת מימד". בני מחלקות מתאימות. הוסיף פונקציית חומם, בני מערך של צורות, מלאו אותו בצורות שונות שלכל צורה ערכים מתאימים והפעיל הפונקציות לוודא שהכל פועל כמורה.

וסףין:

1. בני ArrayList שיכיל את כל הצורות הדו מימדיות (עיגולים ומלבנים). הדפיס את פרטיהם.
2. בני מחסנית שתכיל את כל הצורות העגולות (עיגול, גליל, כדור) הדפיס אותן תוך כדי הוצאה. הדפיס גם את הרדיוס של כל צורה (ההמרה מעצבנת אוتر? ראי פתרון בשאלת 1).
3. בני תור שיכיל את כל הצורות שהיקפן גדול מ50, הדפיס אותן תוך כדי הוצאה.
4. בני רשימה ממוגנת (list sorted) של צורות. הרשימה תטען לפי היקף הצורה.
5. בני רשימה ממוגנת של מלבנים. הרשימה תטען לפי שטח המלבנים.
6. בני MalbenArrayList – מחלקה שיורשת ArrayList ומאפשרת הכנסת מלבנים בלבד (דורשת את כל הפונקציות של ArrayList). הכנסי לתוכה ערכים, נסי להכניס כדור, הדפיס את אורכי כל המלבנים.

802048450



שיעור # 7 - Interfaces

ראשי פרקיים:

★ ממשך – Interface

- ★ ירושה מרובה ממשקים לעומת יחידה מחלקות, היררכית ממשקים
- ★ימוש והמשך – שימוש ממשי וימוש ורטואלי
- ★ ירושת שני ממשקים, בהם פונקציה באותו שם, הכוונה למשק הנוכחי
- ★ שימוש במשקי מערכות
- ★ משתנה מטיפוס המשק,
- ★ פרמטרים מממשק

העומת אופרטורים:

בני פונקציות מתאימות על פי הדרישות הבאות:

- א. שני עיגולים שוים אם ערכי המאפיינים שלהם שווים. כתבי אופרטור == מתאים.
- ב. עיגול שווה לכל צורה אחרת אם שטחים שווה. כתבי פונקציית equals מתאימה.
- ג. חיבור (+) של שני מלבים פירשו מלבן חדש שאורךו סכום שני האורכים ורוחבו סכום רוחבם. כתבי אופרטור + מתאים.
- ד. חיבור (+) של מלבן עם מספר, פירשו הגדלת ערכי המלבן (האורך והרוחב) במספר זה. כתבי אופרטור + מתאים.
- ה. אופרטור מינוס פועל בדרך דומה. כתבי פונקציות מתאימות.
- ו. "מלבן פלוס פלוס" פירשו קידום ערכי האורך והרוחב ב1, כתבי אופרטור מתאים. כתבי דומה גם ל--.
- ז. תיבת גודלה מתייבה אחרת אם נפח הראשונה גדול מנפח השנייה. כתבי אופרטורים <, >, !=, שיאפשרו השוואת בין תיבות.
- ח. תיבת גודלה ממספר כלשהו אם נפחה גדול ממנו. כתבי אופרטורים מתאים.
- ט. כדור מתקיים (true) אם נפחו גדול מ100, כתבי אופרטור true | false מתאים.
- ו. המרה של כדור לעיגול פרושה בנית עיגול ברדיוס ונקודה זהה לכדור. מה יקרה אם תנס' להעמס אופרטור המרה מתאים.
- א. המרה של עיגול או כדור ל string היא החזר ערכי העיגול במחוזת. בני אופרטור מתאים.
- ב. המרה של מחוזת לעיגול היא בנית עיגול ע"פ המימדים שנשלחו. בני פונקציה מתאימה.

שיעור #8 - תוספות

ראשי פרקיים

Garbage collector- GC ★

↳ חלק מהחאים, אחראי על איסוף הצלב,

↳ עוקב אחר אובייקטים שאין להם הפניה בheap,

↳ מפרקן הזיכרון,

↳ מבצע defrag של הזיכרון,

↳ איז מבצע tor'd

↳ הפונקציה dispose – זימון יישיר שלה.

↳ המחלקה GC – הפעלת `garbage`

↳ המתודה collect

↳ GetGeneration

↳ והמתודה GetTotalMemory ↳

תרגילים אפיון מחלקות – תרומות

כתב מחלקות המטפלות בנתונים אודוטות תרומות.

1. לכל תרופה יש לשמר את שמה, את הכמות שלה, את תאריך התפוגה ורשימה של חומרים מהם עשויה התרופה, על כל חומר אלו רציתם לדעת מהו האחד שלו מתוך הרכב התרופה כולה, מהו שמו, וכן האם הוא חומר פועל או לא.
2. כמו"כ לכל תרופה יש את הביעות שבן היא מטפלת. קיימת רשימה קבועה של בעיות (כאב ראש, דלקת אוזניים, דלקת כללית וכו'), תרופה יכולה לטפל ביותר מבעיה אחת.
3. עבור תרומות שהן מסווג סירופ נרצה לדעת גם אתطعم שלו. טעם יכול להיות רק אחד מהערכיטים הבאים: תפוז, תפוח, פטל ופירות.
4. בעת ייצור (בנויות) תרופה מכל סוג יש לקבל את תאריך התפוגה. בעת ייצור סירופ אין חובה לקבל תאריך תפוגה, אך תאריך התפוגה יהיה שלוש שנים לפחות מהיום.
5. לסירופ יש להוסיף פונקציה האם מכיל צבע מאכל? פונקציה זו אינה קשורה לסירופ דווקא, אלא למוצרי מזון נוספים כמו ממתקים.
6. ממש את הפונקציה "האם מכיל צבע מאכל", הפקיעה ותבחן ברשימת החומרים האם יש חומר שמכיל את המילים "צבע מאכל", או "food color".
7. כתבי פונקציית ToString שתחזיר את נתונים התרופה בצורה מחורצת.
8. יש לאפשר בדיקה האם התרופה מותאמת לשימוש בגיל מסוים או לא. באופן כללי תרופה מותאמת לשימוש החל מגיל שנתיים ומעלה, אך סירופ מותר לשימוש כבר מגיל חצי שנה ועוד גיל 12 בלבד.
9. בסירופ יש להגדיל את הכמות ל 250 מ"ל ולא יותר.

משחק זיכרון

להלן תאור המחלקות הנדרשות וחבריהן:

1. כרטיס בסיסי – מחלקה אבסטרקטית

1.1. משתני המחלקה

- האם גלי
- האם כרטיס ראשון או שני

1.2. מתודות המחלקה

- דרישת למתודות Equals, לצורך המתוודה הבאה (וכן לצורך איתחול תקין תוך שימוש בcontains).
- בדיקת התאמה (מקבלת כרטיס נוסף) – מתודה זו תידرس בכל המחלקות היורשות ותשתמש בקודמת.
- ציור הcartis – מתודה זו תידرس בכל המחלקות היורשות, לצורך ציור יחידיות הcartis אך אלגוריתם הציור הכללי יהיה במתודה משותפת.

2. כרטיס סמל

2.1. משתני המחלקה

- סימן (char)
- צבע (ConsoleColor)

3. כרטיס אות

3.1. משתני המחלקה

- אות (char)

4. כרטיס תרגיל

4.1. משתני המחלקה

- תרגיל
- פתרון

5. שחקן בסיסי – מחלקה אבסטרקטית

5.1. משתני המחלקה

- מספר נקודות
- List של כרטיסים

5.2. מאפיינים - Properties

- שם – רק get
- פתרון

5.3. מתודות המחלקה

- איתחול שם – מתודה זו תידرس במחלקות שחקן משתמש בחירת כרטיס להרמה – מתודה זו תידرس בכל המחלקות היורשות הצגת הכרטיסים שברשות השחקן

6. שחקן משתמש

6.1. משתני המחלקה

- שם

7. שחזור מחשב

7.1. משתני המחלקה

- שם השחקן ("מחשב") לקריאה בלבד - קבוע

8. לוח

8.1. משתני המחלקה

- גודל הלוח
- מערך כרטיסי הלוח
- מתודות המחלקה
- אתחול לוח (מקבלת את הכרטיסים האפשריים מהמשחק) – מגילה ומתחלה כרטיסים
- ציור הלוח
- בדיקה אם מיקום כרטיס להרמה הינו חוקי
- בדיקה אם קיימים עדין כרטיסים במשחק
- למתזקדים:
- לוח ג'נרי שמקבל טיפוס כרטיס שבעבורו יבוצע האיתחול,
- אינדקסר לערוך.

9. משחק

9.1. משתני המחלקה

- List של שחוקרים
- Dictionary של כרטיסים קיימים במשחק, המפתח הוא סוג הכרטיס (מונע החנעה של סוג כרטיסים), והערך הוא list של כל הכרטיסים הקיימים בסוג זה.
- אינדקס של השחקן הנוכחי.
- סוג המשחק (ערך מונע ה-*לענונו*)

9.2. מתודות המחלקה

- אתחול שחוקרים והגדירות
- מציאת זוג (מסורת מלהות, מוסיפה לכרטיסי השחקן, מוסיפה נזודות)
- הצגת משתמש המנצח
- מהלך המשחק
 - לולאה שרצה על השחקנים, ובכל סיבוב
 - בחירתן כרטיס (1)
 - בדיקה אם תקין
 - הפיכת הכרטיס
 - בחירתן כרטיס (2)
 - בדיקה אם תקין
 - הפיכת הכרטיס
 - בדיקה אם הכרטיסים הינם זוג
 - אם כן
 - מבצעות ותהליך מציאת זוג
 - בדיקה אם הסתיים המשחק
 - אם כן
 - הצגת המנצח
 - הצגת הכרטיסים שאסף

contains - דרכו את Equals או את ToString הייעזר ב Polymorphism generics - צרי את הלוח עם אפשרות לקביעת טיפוס הכרטיסים. Interface - אפשרי לכל מחלקות הכרטיסים להקריא את עצמן, להציבו. Indexer - הוסיף למחלקה לות. אפשרי למשתמש לבחור האם לקדם בעצמו את המהלך הבא או ע"י המשתמש. FormatException טפל בקלט לא חוקי בעזרת

תרגילי חזרה ל מבחנים

1. שאלות מבחון מתוכן

לפנינו תיאור חלקו של מחלקות, קראנו אותו וענו על השאלות המופיעות אחרי

```

class ProductInRecipy : IHealthMatching
{
    public string Name { get; set; }
    public float Calories { get; set; }
    public string Description { get; set; }
    public List<ProductInRecipy> Products { get; set; }
    public string Instructions { get; set; }
    public int NumberOfPortions { get; set; }

    public Recipy(string recipName)
    {
        Name = recipName;
    }
}

public class Dessert : Recipy
{
    public bool SpecialServingDishNeeded { get; set; }
    public int PricePerServingDish { get; set; }
}

enum eRecommendationLevel
{
    Recommended, Possible, Danger
}

interface IHealthMatching
{
    eRecommendationLevel ForWeightKeepers();
    eRecommendationLevel ForHeartSick();
}

```

1. במחלקה **ProductInRecipy** יש **Properties** הורשים בדיקות תקינות (**Price** חוקי צריך להיות

גודול משקל אחד, **Calories** הוא בין 20 ל 800). רשמי קוד מותאים.

2. למחלקה **Recipy** נדרש להוסיף נתונים המגדיר את הסיווג של המרכיב. קיימת רשימה סגורה

וקבועה של מספר סיווגים: בשרי, חלבני, חמציגי, מהיר וללא גלוטן. כל מרכיב יכול להשתייך למספר

סיווגים. שימושו לב לשימוש בטיפוס הנתונים המותאים ביותר לשימרת המידע הנ"ל.

3. מחיר מנה במתכון מחושב בהתאם למחירי המוצרים וceneותיהם במתכון, וכן מוסף למחיר גם המחיר של כלי ההגשה כאשר הוא נדרש. כתבי קוד מתאים (CalcPrice)

4. כתבי 2 Constructors למחלקה Dessert הראשון קיבל שם מתכון, והשני קיבל שם מתכון, האם נדרש כלי הגשה מיוחד, ומהיר כל הגשה.

5. ברצוננו להוסיף למחלקה מוצר במתכון נתון עזר שיסיע לחשב את הכלים בהן יש להשתמש לפ' הכמות הנדרשת. לצורך זה נוסיף למחלקה אוסף שישמר רשימת שמות כלים ומספר המציג את התוכלה שלהם בגרמים. לדוגמה: כפיט – 8 (גרם), כף – 15 (גרם), כוס – 200 (גרם) וכו'. כמו"כ נוסיף למחלקה מתודת המקבלת כמות של מוצר ושם של כלי מידה רצוי ומוחזירה את הכמות הנדרשת לפי הכלל. לדוגמה: במקרה שהמתודה תקבל כמות = 30 וכלי = "cup" היא תחזיר 2 (כפות), במקרה שהמתודה תקבל כמות = 100 וכלי = "cup" היא תחזיר 1/2 (כוס).
הגדרי את אוסף הכלים בצוורה המתאימה ביותר, וכתבי את המתודה.

6. הגדרי ב Main שני מופעים של מתכנים ואותחלו אותם.

7. מחלקה מתכון אמרה למשמש את המשק התאמה בריאוטית, דאגי שהמחלקה תמשוך אותו כנדרש. ההתאמה לשומר משקל היא לפי כמות הקלוריות במתכון, (מתחת 50 – מומלץ מתחת 200 – אפשרי, אחרית מסוכן). התאמה לחולי לב נקבעת לפי תכולת המלח והשמן (אם יש במתכון גם מלח וגם שמן – מסוכן, אחד משניהם אפשרי, אף אחד – מומלץ).

8. שני מוצרים הם מוצרים שונים אם שם וceneותם זהה, כתבי מתודת Equals מתאימה.

9. בני את מחלקה פעילות גוףנית, המחלקה מכילה את הנתונים: מרחק זמן פעילות (מספר ממשי), דרגת קושי (מספר), תאורה (מחוזת), כמו כן ניתן היה לבדוק לגבי כל פעילות את ההתאמה הבריאותית שלה לבעיות שונות כגון עודף משקל ובעיות לב. ככל שימוש הזמן של הפעולות ארוך יותר היא מתאימה יותר לבני משקל עודף, ככל שדרגת הקושי גבוהה יותר היא מתאימה פחות לבני בעיות לב.

10. קופץ מציגה מידע רפואי לאנשים בעלי בעיות שונות, המידע כולל מתכנים שונים ופעילות גופנית לאורח חיים רפואי. הגדרי את אוסף המידעים בצוורה המתאימה ביותר, ומלאו אותו בשלושה נתונים לפחות. בדק אם האיבר הנמצא במקום השני באוסף מומלץ לחולי לב, אם כן במקרה שמדובר במתכון הדפסי את שמו, במקרה שמדובר בפעולות גופנית הדפסי את תיאורה.

2. שאלות ממבחן בגד

ענין על כל 10 השאלות הבאות! . ($4 * 10 = 40$)

לפניך קוד מחלקת בגד קראו אותו וענין על השאלות שאחריו.

```

class clsClothes
{
    public string shem { get; set; }
    public int achuz { get; set; }

    האחוז של החומר בבגד // }

abstract class clsClothing
{
    public int shem { get; set; }
    public int shnatYizur { get; set; }
    public List<clsChomer> chomarim { get; set; }
    public int price { get; set; }
    public int hanacha()
    {
        if (shnatYizur < DateTime.Today.Year)
            return (DateTime.Today.Year - shnatYizur) * 5;
    }
}

```

מחלקת בגד ניתן לרשות מחלקות רבות למשל: בגדי חורף , בגדי קיץ , בגדי שבת וכו'

אם נבצע (חלוקת) את מחלקת בגדי חורף

שיימי לבן במקורה והתשובה דורשת שינוי/תוספת בחלוקת הבסיס, הדגישי זאת בתשובהך.

שאלות:

1. בחלוקת בגדי חורף ההנחה נקבעת לפי הכלל הבא : אם העונה הנוכחית היא חורף אין הנחה נוספת , בכל עונה אחרת ישנו 10 אחוזי הנחה נוספים מעבר לאחוזי הנחה שמשמעותם בחלוקת בגד.

כתבו את פונקציית חישוב הנחה כאשר חורף ממשועתו חודשיים (3 – 10) וקיז (9 – 4) לצורך חישוב הנחה השימושי בתאריך הנוכחי.

2. כתבו פונקציית הנחה נוספת שהפעם מקבל בתור פרמטר מסוג `sunet` של העונה הנוכחית.

ה`sunet` מוגדר ב `namespace`如下所示的 `eOna{winter, summer}`。

3. בדיקת שעתנד היא פונקציה בוליאנית שחייבת להיכلل בחלוקת בגדי חורף ובחלוקת נוספות. לאחרות למשל ספות, שמיכות וכדומה - מה יש לבצע לשם כך? איר? (כתבו קוד)

4. רשמי את הפונקציה הבוליאנית של בדיקת שעטן – פונקציה זו בודקת באוסף החומרים האם צמר ופשתן כלולים בה – כתבי ביעילות ובקצרה.

5. כתבי מתודה המחזיר את הבגד החול. ביוטר בתוך נוסף מסוג בגדים.

6. ערך בריית מחדל של שנת יוצר של בגד הוא השנה הנוכחית. לבגד חורף יש להגיד בនוסף גם

ערך בריית מחדל למשתנה הבוליאני האם אוטום לגשם? כן.

- היכן תכתב את הקוד? כתבי את הגדירות הנ"ל ללא כפל קוד.
7. כתבי פונקציה Equals הבודקת אם שני בגדים הינם שווים, הבגדים שווים אם יש להם אותו שם ואותה שנת ייצור.
8. הגדרי ב Main שלושה בגדים מסוגים שונים ותחליל את משתני המחלקה שלהם.
List<clsClothing> Is; המונח הגדרת אוסף בגדים – ההגדרה כתובה בחו"מ. – מה עלי הניח כי האוסף הת מלא בפרטי לבוש בהתאם לבחירת המשמש
9. בדק אם בגדי שלישי באוסף יש שעטנז. אם יש בו שעטנז מחק אותו מהאוסף.
10. ברצוני למיין את אוסף הבגדים לפי מחיר. המון יבוצע בעזרת הפקודה ()Is.sort(). – מה עלי לבצע לשם כך? (כתב קוד)

3. שאלות מבחן חייו

חלק א – מחלקות

תאור כללי:

המשימה הנדרשת היא בניית מחלקה חיה בעלת התכונות הבאות: מספר החיה, שם החיה, סוג המזון אותו היא אוכلت, גיל ומשקל.

למחלקה יהיו היכולות הבאות (פונקציות): הדפסת תכונות החיה, החזרת מחרוזת של תוכנות החיה, האכלת החיה (שתי וריאציות) ו חישוב כמות המזון אותו החיה אוכلت.

פרוט

1. בני מחלקה בשם Animal.

בנייה התכונות.

2. בני את התכונה משקל.

הגדיר מופע בחוון והשתמשו בו

3. מספר סידורי – כל מופע יקבל מספר אוטומטי. המופע לא יוכל לשנות ערך זה. כיצד תגדיר תכונה זו?

הגדיר מופע נוסף בחוון והדפיס את מספרו. מה יהיה ערכו בהנחה משתני ה private נשלחו?

הין תאתחל משתנים אלו ויכיזב?

הגדיר מופע נוסף בחוון והדפיס את מספרו. מה יהיה ערכו בהנחה שסעיף 2 בוצע?

4. סוג המזון – סוג המזון הוא טקסט חופשי.

כתבו תכונה מתאימה.

מבנה פונקציות

5. כתבו בונה ריקה (התיחסו לקיים בונה נוסף)

כתבו שליטה בחוון Main

6. כתבו בונה נוסף המקבלת ערכים לתכונות השונות.

כתבו שליחה לבונה זו בחוון

7. כתבו פונקציה print שהמדפיס את תוכנות המחלקה.

8. כתבו פונקציה ToString אשר תחזיר מחרוזת של תוכנות המחלקה. (זכיר כי זו פונקציה קיימת במחלקה Object). (במקרה של String toString()

כתבו פונקציית האכלת – הפונקציה מדפסה : "אני אוכל את ..." (סוג המזון המופיעים של החיה).

9. כתבו פונקציה האכלת מועמסת לפונקציה ותקודמות. הפונקציה מקבלת כפרמטר סוג מזון ומדפסה את המשפט "אני אוכל את ..." (סוג המזון שנשלח לפונקציה).

10. כתבו פונקציה שתחזר את כמות האוכל שעל החיה לאכול. (לשם דוגמה: המשקל חלק חמש)

חלק ב' הורשות וממשקים

מתואר מבנה המחלקות הבא:

חיות מים – חיית מים היא חייה אשר ידוע עליה גם עומק המים בו היא חייה.

עופות – כל העופות הם חיים החיבוט לעוף כאשר ידוע עליהם גם מהירות תעופה.

שרצים – הם חיים אשר חיבוט לעוף.

1. השלימי את המילים החסרות בסכמתה הבאה:

waterAnimals: _____

{

}

baleykanaf

{fly();

}

flies: _____

{

}

shrazim

{

}

2. הוסיף את התוכנה עומק המים. ערך התוכנה יכול להיות ردוד, בינוני, عمוק.

היכן תגדיר את `howDeep`?

כיצד תגדיר את `howDeep`?

כיצד תגדיר את התוכנה כולל משתנה ה `private` שלה?

3. כתבי מחדש את פונקציית `print`.

4. חיות מים אוכלת פחות מאחר והמים הם חלק ממדונה.

במקרה והמים רדודים 10% פחות, מים ביןוניים – 20%, מים עמוקים – 30%

כתבו את פונקציית כמות האוכל מחדש בהתאם לכליים אלו.

נסו לכתוב את הפונקציה ללא `if`, (רמז: היעזרו ב `sum`)

5. כתבו בונה חדשה למחלקה שבנית.

6. כתבו את הפונקציה `flying` ב (מחלקת? ממשך?) `fly`.

7. אלן פונקציות חייבות לכתוב במחלקת `flyings`.

(חייבות מבחינת המבנה ולא מבחינה התוכן.)

8. הגדרו בחומר מערך של חיים. הציבו לתוךו מופעים מכל הסוגים: עופות, עופות מים, חיים,

רגליות וכו'.

9. רצוי על המערך והדפסי את תוכן.

את איזה פונקציית הדפסה הוא מפעיל? למה?

10. אם מזכיר בחייב מיט(הנמצאות בתוך המאגר) התייה במים עמוקים יש להדפיס את הודעה הבאה:

"החיה חייה במים עמוקים"

11. ישנו שומר כללי אחד לכל גן החיות היכן ותגידי אותו וכייז?

12. לכל סוג חייה (חייה, עוף מים, שרכים) יש אדם שתפקידו לאכילה. אותו אדם אחראי על כל החיות

מאותו סוג.

היכן וכייז תגידי אותו?

בהתאם

לעכבר קוראנו מילון ערך אונליין
ולא מילון

4. שאלות מבוחן תרופה

```

class Material {
    public int Percent { get; set; } // אחוז החומר
    public string Name { get; set; } // שם החומר
}

abstract class Medicine {
    public string Name { get; set; }
    public float Quantity { get; set; } // תכונה
    public DateTime ExpirationDate { get; set; } // תאריך תיפה
    public Material[] Materials { get; set; }
}

```

נתונה מחלקה תרופה, מחלקת זו ניתן לרשף מחלקות רבות, לדוגמה: גלולות, סירופ, משחה ועוד. אנו נטפל במחלקה תרופה הבסיסית ובמחלקה סירופ.

שימי לב! במקרה והתשובה דורשת שנייה/תוספה במחלקה הבסיסי, הדגישי זאת בתשובתך.

1. בעת ייצור (בנייה) תרופה מכל סוג יש לקבל את תאריך התיפה. בעת ייצור סירופ אין חובה לקבל תאריך תיפה, אז תאריך התיפה יהיה שלוש שנים מהיום.
2. לסירופ יש להגדיר תכולה מקסימלית שמתארת את הכמות המקסימלית שניתן למלא בסירופ מכל סוג שהוא. בעת עדכון התכולה שלא עלתה על התכולה המקסימלית, הגדרי את הנתון תכולה מקסימלית וכן הוסיף את בדיקת התקינות במקום מתאים.
3. הוסיף נתון המכיל את מספר התרופות שייצרנו בחברה, דאגי לכך שהנתון יחשב אוטומטית לעלה עם כל ייצור של תרופה נוספת.
4. הוסיף את הנתון "טעם" לסירופ, טעם יכול להיות רק אחד מהערבים הבאים: תות, תפוז, פטל ופירות. כתבי את כל החלקים הנחוצים להגדרת משתנה מסוג מתאים.
5. לסירופ יש להוסיף פונקציה האם מכיל צבע מאכל? פונקציה זו אינה קשורה לסירופ דווקא, אלא למוצר מזון נוספים כמו ממתקים.
6. ממשי את הפונקציה "האם מכיל צבע מאכל", הפונקציה תבדוק בראשימת החומרים האם יש חומר שמכיל את המילים "צבע מאכל" או "color food".
7. כתבי פונקציית ToString שתחזיר את נתונים מחלקה תרופה בצורה מחורצת.
8. הוסיף פונקציה המתקבלת גיל ומחזירה ערך בוליאני האם התרופה מותרת לשימוש בגיל זה או לא. באופן כללי, תרופה מותרת לשימוש החל מגיל שנתיים ומעלה, אך סירופ מותר לשימוש כבר מגיל חצי שנה ועד גיל 12 בלבד.
9. הוסיף ב main מערך מסוג Medicine ואתחלי אותו בשתי תרופות. אתחלי את כל נתונים התרופות, תרופה אחת אתחלי בקוד ע"י רשיימת אתחול (תחזיר מקוצר), ואחד ע"י קלט מהמשתמש.
10. כתבי פונקציה מקבלת את המערכת הנ"ל ומחזירה במערך חדש רק את התרופות שאין בהן צבע מאכל.

5. מבחן מסכם בשפט #א- (חומר סגור)

חלק א' - עני כו/לא

ענין על 18 מתוך 22 (36=18*2)

1. האם משתני private מחלוקת הבסיס קיימים בחלוקת הירושת? כן
2. האם פונקציה סטטית חייבת להיות בחלוקת סטטית? כן
3. האם פונקציה אבסטרקטית חייבת להיות בחלוקת אבסטרקטית? כן
4. האםחלוקת אבסטרקטית חייבת להכיל פונקציה או property אבסטרקט? כן
5. האםחלוקת סטטית חייבת להכיל חברחלוקת סטטי? כן
6. האםחלוקת סטטית יכולה להכיל חברחלוקת שאינו סטטי? כן
7. האםחלוקת אבסטרקטית יכולה להכיל פונקציה וירטואלית? כן
8. האם ניתן להגדיר תוכנה בתוך משקל? כן
9. האם ניתן להגדיר אובייקט מסווג שלחלוקת אבסטרקטית? כן
10. האם כאשר ctor אחד מפנה ל ctor אחר, ה ctor הקורא יבצע אחרון? כן
11. האם המשפט הבא הוא חוקי: Base b = new Derived(); כן
12. האם בשימוש ב overloading מבילים בין המתודות ע"י הערך המוחזר? כן
13. האם מכלחלוקת יכול לרשעתחלוקת אחת ורבה ממשקי? כן
14. האם בעט משתנה מסווג enum(enum) נשמער בזיכרון C string המכיל את שם הערך בעט enum? כן
15. האם מחלוקת נזנות מכירה יכולה לגשת לכל מרכיביחלוקת הבסיס פרט ל Properties שלה? כן
16. האם מתודה וירטואלית חייבת להיות ברמת הרשאה של protected לפחות? כן
17. האם המתודה RemoveAt מקבלת כפרמטר אובייקט ומיסורה אותו מהאוסף? כן
18. האם שימוש באוסף גנרי עשוי למנוע שגיאות זמן ריצה? כן
19. האם באוסף גנרי מסווג Employee ניתן להכט אובייקטים מסווג Manager? כן
20. האםחלוקת שלא הוגדרה לה בונה בקוד ניתן להקצת מופע? כן
21. האם בתוךחלוקת שמודדרת C sealed יש הבדל בין הרשות ל protected בעבר? כן
22. האם ניתן להגדיר במשקל חבר סטטי? כן

חלק ב' – סמני את התשובה הנכונה (רק תשובה אחת נכון בכל שאלה)

ענין על 8 מטרס 10 ($2^8 = 16$)

1. בעת ביצוע הידור של קובץ הכתוב בשפת C#

- א. יוצר קובץ ביןארי
- ב. יוצר קובץ שמכיל קוד בשפת JMS
- ג. סוג הקובץ שיוצר תלוי בסוג מערכת הפעלה שעליה מבצעים את הידור
- ד. יוצר קובץ שאינו מקומפל, הקומpileציה תבוצע רק בזמן הרצת התוכנית.

2. נתונה מחלוקת מלבן המוגדרת כר' :

Class Malben

```
{  
    Point p;  
    Int x, y;  
}
```

- א. מחלוקת מלבן מכילה מופיע של מחלוקת נקודה
- ב. מחלוקת מלבן מכילה ייחוס (מצבי) של מחלוקת נקודה
- ג. מחלוקת מלבן יורשת מחלוקת נקודה
- ד. נקודה היא מחלוקת המוכלת בתוך מחלוקת מלבן

3. לאלו מחברי מחלוקת נקודה ניתן לגשת מתוך מחלוקת מלבן

- א. רק לאלו שיש להם הרשות public
- ב. מכיוון שהנקודה נמצאת בתוך המלבן. ניתן לגשת לחבריו מחלוקת נקודה מכל ההרשאות
- ג. רק לאלו שיש להם הרשות public או protected
- ד. מחלוקת מלבן אינה יכולה לגשת לחבריו מחלוקת נקודה, ניתן לגשת לחבריו מחלוקת נקודה מה Main בלבד.

4. במילת המפתח new

- א. חובה להשתמש במילה זו בכל הגדרת משתנה מכל סוג שהוא.
- ב. חובה להשתמש במילה זו בהגדרת אובייקטים
- ג. חובה להשתמש במילה זו בכל הגדרת members data
- ד. אין חובה להשתמש במילה זו.

5. כדי שמחלקה יורשת תוכל להשתמש במשתנה שהוגדר במחלקה האב צריך להגדירו כ:

- א. Private
- ב. Protected
- ג. לא משנה
- ד. Const

6. מה מבצעת הקריאה `cls []arr=new cls[10]`

- א. מקצה מערך של 10 אובייקטים מט' פוי cls
- ב. מקצה מערך של 10 אובייקטים מט' פוי arr

ג. מקצתה מערך של 10 מצביעים לאובייקטים מסווג als - כ' ו' נז'ן

ד. מקצתה מערך של 10 מצביעים לאובייקטים מסווג arr

7. לאחר ביצוע הагדרות הבאות : int i c als, איזה מהמשפטים נכון?

א. קיים משתנה מטיפוס int ומשתנה מטיפוס als

ב. קיים מצביע מטיפוס int ומצביע מטיפוס als

ג. קיים מצביע מטיפוס int ומשתנה מטיפוס als

ד. קיים משתנה מטיפוס int ומצביע מטיפוס als

(ד)

8. בסופו מספר עקרונות חשובים:

א. פונקציות , אובייקטים , מחלקות

overriding , overloading , Oop

encapsulation , Polymorphism, inheritance (ג)

subclass | base class

9. איזה מהמשפטים הבאים נכון:

א. בכל מחלקה חובה להגדיר פונקציה toString

toString את פונקציית (ב)

בכל מחלקה אפשר (וכדי) לדרשו את פונקציית toString

המחלקה object מורישה את הפונקציה לכל המחלקות אך אין סיבה לכתוב אותה

ד. המחלקה object מורישה את הפונקציה למחלקות שרצות

10. נתון הקוד הבא:(
Employee e = new Manager();

יכיזד ניתן לגשת לתוכנה בונוס של המנהל?

א. E.Bonus = 100

.(e as Manager).Bonus = 100 (ב)

(Employee as Manager).Bonus = 100 (ג)

ד. לא ניתן לגשת לתוכנה בונוס

חלק ג' - תרגמי את הודעות השגיאה הבאות והסבירו אותן, כמו כן כתבי כיצד ניתן לתקן את השגיאה.

עב' על 4 מטר 6 ($8=4*2$)

Employee.salary is inaccessible due to its protection level .1

תרגום: השגיאה היא של המוליך של salary לא ניתן לגשת לו
תיקון: השגיאה ניתן לגשת לו

Class Derived does not implement inherited abstract member base.Func .2

תרגום: השגיאה היא של המוליך של Func לא יונקן בderived
תיקון: השגיאה יונקן בderived

Class Person has no constructor that takes 2 arguments .3

תרגום: השגיאה היא של המוליך של Person לא ישנות בarguments 2
תיקון: השגיאה ישנות בarguments 2

Cannot create an instance of the abstract class or interface shape .4

תרגום: השגיאה היא של המוליך של interface לא ניתן ליצור ערך
תיקון: השגיאה ניתן ליצור ערך

Employee.CalcBonus(): no suitable method found to override .5

תרגום: השגיאה היא של method CalcBonus() לא ניתןoverride
תיקון: השגיאה ניתןoverride

Property or indexer Tree.Age cannot be assigned to. It is read-only .6

תרגום: השגיאה היא של property Age לא ניתן לassign
תיקון: השגיאה ניתן לassign

כל הבדיקות

1. בוחן 1

א. סובייכט או לא סובייכט:

1. הפקודה `a = new Account()` יוצרת אובייקט מסוג Account
2. ב `C#` כל האובייקטים מוקצים דינמיות
3. ב `C#` ניתן לכתוב קוד בתוך המחלקה או (מחוץ למחלקה)
4. ההגדירה `[10] int[] arr = new int[10]` מגדירה מצביע לערך ו-10 מצביעים ל `int`
5. ההגדירה `[10] Person[] p = new Person[10]` מגדירה מצביע לערך `p` ועשרה מצביעים לסוג `Person`
6. השתמש ב `num` כאשר יש לו רשיימה של ערכים.
7. הרשאות public מאפשרות גישה מתוך המחלקה או מהמחלקות היורשות

ב. מהם 3 עקרונות של תיכנות מונחה עצמים – תרגימי והסבירו:

8. Encapsulation - גיבוב
9. Inheritance -iritance
10. Polymorphism - פולימורפיזם

ג. הודעת שגיאה

Object reference not set to an instance of an object. הסבירו את הודעת השגיאה הבאה: `Le résultat de l'appel à la méthode ToString() sur null n'est pas autorisé.`

2. בוחן 2 (10 מתקן 13)

א. סובייכט או לא סובייכט

1. כל הקזאה של אובייקט (שאינו ממחלקה יורשת) מפעילה תמיד בנאי אחד.
2. הקזאה של אובייקט ממחלקה יורשת ממחלקה אחרת מפעילה לפחות שני בנאים.
3. הפקודה `[5] Product[] arr = new Product[5]` מפעילה את הבניי של מחלקה `Product` 5 פעמיים.

4. הפקודה `[5] arr = new Product[]` מפעילה את הבנייה של מחלקת `Product` 5 פעמים.
5. אובייקט מחלוקת יורשת מכל את משתני ה `Private` של מחלוקת הבסיס.
6. מחלוקת הבסיס יכולה לחשוף למשתנים בהרשאת `Protected` שנמצאים בחלוקת היורשת.
7. `Property` מכל בתוכו את הערך שהציבו בו.
8. לכל `Property` חייב להיות `get` ו- `set`.
9. מ' `the properties` וה `fields` חייב להיות שווה באותה מחלוקת.
10. מחלוקת שולחן יורשת מחלוקת רהיט.
11. מחלוקת מרפאה יורשת מחלוקת רופא.

ב. הסביר את השגיאה והציע תיקון:

12. Object Reference not set to an instance of object
 13. An unhandled exception of type 'System.StackOverflowException'
- occurred
occurred after some time
occurred after some time
3. בוחן 3 (10 מתוך 12)

א. סמני נכון או לא!

1. חובה לדرس בחלוקת היורשת כל פונקציה שהוגדרה כירטואלית בחלוקת הבסיס.
2. למערך מסווג פרח ניתן להכניס פרחי חממה?
3. למערך מסווג עץ ניתן להכניס צמחים?
4. הפונקציה הדורשת חייבת להתאים לפונקציה שבחלוקת הבסיס בטיפוס הפרמטרים ובכמותם, אך לא בערך המוחזר.
5. כל פונקציה דורשת חייבת להפעיל גם את הפונקציה הבסיסית ע"י קריאה ל `base`.
6. כל פונקציה דורשת פעולה אוטומטית גם את הפונקציה בחלוקת הבסיס.
7. אם כתבנו על פונקציה `override` היא תופעל גם אם המצביע של האובייקט הוא מחלוקת הבסיס.

8. אם כתבנו על פונקציה new היא תופעל גם אם המצביע של האובייקט הוא מחלוקת הבסיס.

9. כישיש מצביע מחלוקת הבסיס שצביע על אובייקט מחלוקת יורשת נוכל לגשת דרכו רק לחבריה המוגדרים בחלוקת הבסיס.

10. אם נרצה לגשת מטור מחלוקת מבצע לשנתנה המוגדר בחלוקת מוצר נצטרך לתת לו הרשאה של Protected.

ב. הסביר את השגיאה והציעי תיקון:

- cannot override inherited member

'Lesson_2__Inheritance.Person.Func2()' because it is not marked virtual, abstract, or override

- Cannot implicitly convert type 'Plant' to 'Tree'. An explicit conversion exists

בוחן 3#

1. המשטנים הבסיסיים והאובייקטים עוברים לפונקציות **value by value** כברירת המחדל.

2. משתנה מסווג int עובר **by reference** כברירת מחדל.

3. כאשר אנו מעבירים משתנה, משתנה by reference, על המשתנה המקורי מועבר לפונקציה, ולא העותק שלו. כתוצאה לכך, אם נשנה את הפרמטר במהלך הפונקציה, ישנה גם המשתנה מחוץ לפונקציה.

4. אין צורך לאותחל את המשתנים הנשלחים או כמשתני ref

5. בסיום ריצת הפונקציה, משתני Out חייבים להכיל ערך כלשהו.

6. **משתנים סטטיים** אלו משתנים השייכים למחלוקת ולא לאובייקט מסוים שלה, וכן הם משותפים לכל האובייקטים של אותה מחלוקת.

7. ניתן לאותחל את המשתנה הסטטי ורק מיד עם הגדרתו או **בבנייה בורית** המחדול.

8. הפונקציה **הסתטית** תיקרא לפני יצירת האובייקט הראשון של המחלוקת.

9. הגישה ל **const** נעשית בעזרת שם המחלוקת.

10. משתנה המוגדר כ **readonly** - ניתן לאותחל רק בתור המחלוקת.

.11. כמה מצביעים וכמה אובייקטים ממחלקה Person הוקזו בקוד הבא?

Person p = new Person();

p = new Person();

Person [] arr = new Person[5];

arr[0] = p;

א. 7 מצביעים ו 2 אובייקטים

ב. 7 מצביעים ו 7 אובייקטים

ג. 6 מצביעים ו 2 אובייקטים

ד. 6 מצביעים ו 7 אובייקטים

.12. כאשר פונקציית params:

היא לא יכולה לקבל פרמטרים נוספים

אם יש פרמטרים נוספים הם כתבו אחרי הparams

" " " " " לפני "

.13 static, const

שניהם בرمת המחלקה, Const משטנה static לא

אל מולדים נרדפות

שניהם בرمת המופיע, const לא משטנה static כן

" " " " " המחלקה , "

: out, ref .14

לא מוגדר עבורם שטח בזיכרון של הפונקציה

מוגדר שטח שמייחס לכתובות הזיכרון של המשתנים

" " שיש בו העתק של הערך שהתקבל מהתוכנית

.15 תרגמי את המושגים המודגשים (6 מתוך 8)

4. בוחן

תاري את הדרך לתקן: (תרגום: 2 תיקון: 1.5. 21 נקודות 3 שגיאות רשות)

1. Inconsistent accessibility: base class 'ConsoleApplication2.Plant' is less accessible than class 'ConsoleApplication2.Tree'

43

2. `Plant.Details()'`: virtual or abstract members cannot be private

~~WE CAN'T USE PROTECTED FOR ALL PUBLIC/PRIVATE MEMBERS AS DETAILS IS INHERITED~~

3. '`ConsoleApplication2.Plant.identity`' is inaccessible due to its protection level

~~PROTECTED IS PRIVATE WHICH MEANS ANY MEMBER EXCEPT IDENTITY CAN'T ACCESS IT~~

4. (Warning) '`ConsoleApplication2.Tree.Details()`' hides inherited member

~~THIS IS A MEMBER~~

~~AS IT IS A MEMBER OF PLANT AND TREE HAS THIS MEMBER~~

~~MEMBER WHICH IS NEW MUST BE OK~~

5. '`ConsoleApplication2.Plant.Details()`'. To make the current member override that implementation, add the `override` keyword. Otherwise add the `new` keyword.

~~IF WE DON'T USE OVERRIDE, IT OVERLOADS THE OTHER MEMBER~~

~~NEW OR OVERRIDE MUST BE USED~~

6. (Warning) The field '`ConsoleApplication2.Warm.WarmName`' is never used.

~~IT IS NEVER USED - SO IT IS OK~~

~~SO IT IS OK~~

7. Keyword 'this' is not valid in a static property, static method, or static field initializer

~~IT IS NOT VALID IN STATIC PROPERTY, STATIC METHOD OR STATIC FIELD INITIALIZER~~

~~STATIC OR THIS IS ERROR IF~~

8. A readonly field cannot be assigned to (except in a constructor or a variable initializer)

~~READONLY FIELD CAN'T BE ASSIGNED TO~~

~~EXCEPT IN CONSTRUCTOR OR VARIABLE INITIALIZER~~

9. Cannot create an instance of the abstract class or interface

`'ConsoleApplication2.Plant'`

~~ABSTRACT CLASS CAN'T BE INSTANTIATED~~

~~ABSTRACT CLASS CAN'T BE INSTANTIATED~~

10.'ConsoleApplication2.Tree.Tree()': access modifiers are not allowed on static constructors

הצורה היגיינית היא `public static Tree()` ו-`static` מותר בפונקציית `Tree` אך לא בקונסטרוקטור.

5. בוחן 5- הורשה ורב צורתיות

נכון או לא נכון ($17^2 = 289\% = 34\%$):

מאפיינים

- מנגןון ה **properties** - מחייב להגדיר גם משתנה חבר מחלוקת פרטி לכל **property**.
- מנגןון ה **properties** - מאפשר לנו צורת עבודה המזיכירה עבור משתמש הקצה עבודה מול שדות יישרות, כאשר בפועל יקרו פונקציות `get` ו-`set` - עם כל קראיה של המשתמש.
- מנגןון ה **properties** - מחייב להגדיר גם `get` וגם `set`.
- מאפיינים אוטומטיים מאפשרים לגשת למשתנה פרטֵי שמתחליל באוט קטנה, שנוצר במרומץ.
- בגוף מתודות `get` וה-`set` לא נפנה לחבר המחלוקת הפרטֵי אלא רק לציבורו.
- הערך `value` מכיל למעשה את מה שהמשתמש כתוב לאחר ההשמה.

הורשה

- מחלקה היורשת ממחלוקת אחרת, יכולה לגשת גם אל שדות ה `public` - `שלה`, וגם אל שדות ה `private`.
- כשכתבנו במחלוקת נגזרת (`x new public void SetX(int)` כתוב `new` החבamo פונקציה עם שם `X` של מחלוקת הבסיס).
- הרשאת `protected` חשפת את משתנה המחלוקת רק למחלוקת היורשת, למחלוקות חיצונית לא תהיה גישה למשתנה כלל.
- כדי למנוע ירשה ממחלוקת נגדיר את חברי המחלוקת `protected` ואת המחלוקת `sealed`.

פולימורפיזם

- ניתן למשמש **Polymorphism** גם ללא **Inheritance** ואין בהכרח קשר

הדוק בינויים

12. ייחוס מהמחלקה הנגזרת יכול להתיחס (להציביע) על אובייקט מחלקת הבסיס.
13. ייחוס מטיפוס הבסיס יכול להתיחס (להציביע) על אובייקט מחלקת הנגזרת.
14. כדי שהמחלקה הנגזרת תוכל **לדרוס**/לرمוס את המימוש המקורי במחלקת הבסיס, עליה להשתמש בשם מתודה זהה ובמילה **Override**, אך **טיפוסי** הפרמטרים והערך המוחזר. יכולים להיות שונים.
15. אין שום סכנה בביצוע המירה מחלקת הנגזרת למחלקת הבסיסית וניתן לבצע אותה באופן **מורםץ**.
16. מחלוקת **מופשטת** יכולה עצמה לרשף מחלוקת רגילה או מחלוקת מופשטת אחרת.
17. חייבים למשם מתודות מופשטות במחלקה הנגזרת.
18. מתודות מופשטות הינן וירטואליות.

6. בוחן

תרגמי ותקני: (18%)

- 'shape.Shape.MyType()' is abstract but it is contained in non-abstract class

- 'shape.Triangle' does not implement inherited abstract member 'shape.Shape.MyType()'

- 'shape.Ball' does not implement interface member 'shape.I3D.CalcShetachPanim()'

- interface I3D:Shape => Type 'Shape' in interface list is not an interface

5. Circle c;

```
if (arr[i] is Circle)
c=arr[i];
```

אנו מודים לך על תשובתך!

Cannot implicitly convert type 'shape.Shape' to 'shape.Circle'. An explicit conversion exists (are you missing a cast?)

-
6. A local variable named 'c' cannot be declared in this scope because it would give a different meaning to 'c', which is already used in a 'parent or current' scope to denote something else
-

7. return true;

return false;}=> Warning:Unreachable code detected

ציני נכון או לא נכון ותקני כשאינו נכון: (22%)

1. בשפת # C מחלקה יכולה לרשת מחלקה אחת בלבד + ממוק אחד בלבד.
2. ממוק הינו מבנה לוגי שמכיל רק הצהרות.
3. ממוק אינו יכול להכיל מתודות עם מימושים או .Data Members
4. ממוק דומה למחלקה אבסטרקטית בזה שלא ניתן לייצר ממנו מופעים.
5. ממוק דומה למחלקה אבסטרקטית בזה שהוא יכול להגדיר רק חברים מופשטים.
6. ממוק דומה למחלקה אבסטרקטית בזה שהמימושים של מתודות הממשק במחלקות הנגזרות חיבים להיות וירטואליים.
7. בממוקים לא ניתן להגדיר הרשות גישה.
8. כל הרשות הגישה של הממשק הן Protected
9. ממוק יכול לרשת מחלקה אחת ורבה ממוקים.
10. מצבייע מסווג הממשק יוכל להכיל הפניה למחלקה שמיימה את הממשק.
11. המתודה ToString שמחזירה מחרוזת המתארת את האובייקט, קיימת לכל האובייקטים בדוט נט.
12. המתודה Equals מבודעת השוואה בין שני אובייקטים . כבירית המחדל הפונקציה מבודעת השוואה של ערכי התו"חשות בלבד, כלומר מחזירה true אם המשתנים מצבעים אל אותו אובייקט אולם ניתן בכל מחלקה לדרכו אותה כך שתשווה לפי מאפיינים אחרים של האובייקט

ולפ"ע שם: זיו ציון:

בחן #2

1. איך נקראת הפונקציה שהיא נקודת הכניסה לתוכנית? main ✓
2. אילו 2 הרשות גישה קיימות ב-#? private public ✓
3. איך נקרא המשטנה הנוסתר שנשלח לכל מתודה שהופעלה ע"י מופע set this ✓
4. מה המילה שעל ידה תופס האובייקט מקום פיזי בזיכרון? New ✓
5. מה מכילה כל מחלקה? data member members ✓
6. `Console.WriteLine(P.Width);` סמנטי נכון או לא נכון? לא נכון ✓
7. הפונקציה הנ"ל מודפסה למסך את P.Width ✓
8. הפונקציה הנ"ל קוראת קלט מה משתמש ומכניסה אותו לתוך Width ✓

א. בחרו בתשובה הנכונה:

7. ההבדל בין `enum` ל `Flags Enum` הוא:
ב `enum` א"א לשנות את מספרי הפריטים וב `Flags Enum` אפשר
ב. `Flags Enum` מאפשר בחירה מרובת
ג. `enum` נשמר כ `int` | `enum` `Flags Enum` `byte`
8. כמה מצביעים וכמה אובייקטים ממחלקת `Person` הוקזו בקוד הבא?

`Person p = new Person();` –

`p = new Person();` –

`Person [] arr = new Person[5];`

`arr[0] = p;`

ב. arr[0] = p;
arr[0] = arr[1]
= 6

9. האם `Console.WriteLine` מחלקה, משתנה חבר מחלוקת, מרחב שמות, מתודה או טיפוס?
10. האם `System` הוא מחלוקת, משתנה חבר מחלוקת, מרחב שמות, מתודה או טיפוס?
11. האם ניתן ליצור מתודה מחוץ למחלוקת? כן/לא ✓
12. האם משתנה רשימה `enum` הוא `reference type` או `value type` ✓
13. האם משתנה מערך שוקלים הוא `reference type` או `value type` ✓
14. האם כל איבר במערך הנ"ל הוא `RT` או `value type` ✓
15. כאשר רשימתרכיבי השוקולד הינה מערך מחרוזות - האם המאפיין-Ingredients הינו `RT` או `value type` ומה לגבי כל איבר במערך * האם הינו `reference type` או `value type` ✓
16. כאשר רשימתרכיבי השוקולד הינה מערך אובייקטים - האם המאפיין-ChocolateIngredients הינו `RT` או `value type` ✓
17. לפי 5 תשובותיך. הקודמות - אלו מהמקרים הנ"ל מחייבים שמי לפני הגישה לערך שלהם?
18. האם כאשר נקצתה 2 חבילות נופש המצביעות לאותו מלון- האם יוצרכו 2 אובייקטים מסווג מלון? ✓
19. כאשר מוסיף שוקולד לרשימה של שוקלים שבמערך - בהכרח שהוא יכול גם את אותה רשימתרכיבים?
20. כאשר נשנה איבר ברשימה של הרכיבים המשותפת - לכל גדי השוקולד-פרווה. נctrkr לשנות אותו בנפרד גם לכל אחד מה אובייקטים שוקולד-פרווה? –

ב. הודיעת שגיאה הסבירי את הודעות השגיאה הבאה ותקני:
הכוון לאירועים (ב) אוסף גיבובים (ב) אוסף גיבובים (ב) אוסף גיבובים (ב)
(ב) אוסף גיבובים (ב) אוסף גיבובים (ב) אוסף גיבובים (ב)

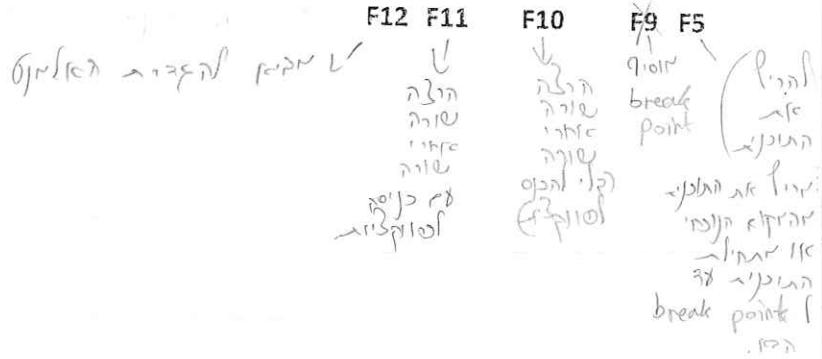
Object reference not set to an instance of an object
The type or namespace name 'Person' could not be found (are you missing a using directive or an assembly reference?)

הכוון לאירועים (ב) אוסף גיבובים (ב) אוסף גיבובים (ב) אוסף גיבובים (ב)
Person p = new Person();
Person Person Person Person

- 'Application2.Person.i' is inaccessible due to its protection level
- Use of unassigned local variable 'p'

בצלחה!!

ג. מה מבצע הלחצן הבא:



price pric

private double price;

public double Price

{ get { return price; }

set { if (value > 1) price = value; }

}

(7)

private int calories;

public int Calories

{ get { return calories; }

set { if (value >= 200 & value <= 800) calorie = value; }

}

(8)

[Flag] enum eKind{ meat, milk, party, quickly, noGluten }

public ekind kind {get; set; }

(9)

public double calcPrice()

{ public double sum {get; set;}

foreach (var item in Products)

{ sum += item.price * item.Amount;

if (this is Dessert) (this as Dessert).ssdn)

{ sum += (this as Dessert).PPSD; }

}

return sum;

public Desert(string Name)

{ Name = name;

}

(10)

public Dessert(string name, bool ssdn, int ppsd)

{ Name = name; Dessert (name);

SSDN = ssdn;

PPSD = ppsd;

}

-1017-

```
public Dictionary<string, int> List = new Dictionary<string, int>()
{
    spoon: 15,
```

1

```
public double change (String Dish, int Amount)
{
    return Amount / List[Dish];
```

```
Dessert d1 = new Dessert("IceCream", true, 10);
```

(6)

```
Dessert d2 = new Dessert("Cake");
```

```
eReco... ForWe...()
```

(F)

```
{ if ( calories < 200 )
    return eReco... . Recommend;
```

```
else if ( calories < 200 )
    return eReco... . Possible;
```

else

```
    return eReco... . Danger;
```

3

```
eReco... ForHeart...()
```

```
{ public int flag = 0;
```

```
foreach (var item in Products)
```

```
{ if (item.name == "P" || item.name == "S"))
    flag++;
```

```
if (flag == 0)
```

```
    return eReco... . Recomend;
```

```
if (flag == 1)
```

```
    return eReco... . Poss;
```

```
return e... . Pan;
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Runtime.InteropServices;
5 using System.Text;
6 using System.Threading.Tasks;
7
8 namespace Struct
9 {
10
11     class Program
12     {
13         public static void funcWithOptionalParameter(int i, int x=1000, int
14             y=2000)
15         {
16
17         }
18         //out
19         // מאפשר 'להחזיר' יותר מערך אחד מהפונקציה out משנה מסוג
20         // כיוון קיבל ערך במהלך הפעולה Out משנה
21         // גם לפני הפעלה וגם לפני הארגומנט out מצינים//
22         // בשליפה וגם בקבלה
23         static bool func(out int oValue)
24         {
25             oValue = 7000;
26             return true;
27         }
28         //ref
29         static bool funcWithRef(ref int oValue)
30         {
31             oValue = 7000;
32             return true;
33         }
34         public static int SumWithParams(params int[] par)
35         {
36             int sum=0;
37             foreach (var item in par)
38             {
39                 sum += item;
40             }
41             return sum;
42         }
43         public static int SumWithoutParams(int[] par)
44         {
45             int sum = 0;
46             foreach (int item in par)
47             {
48                 sum += item;
49             }
50             return sum;
51         }
52         static void Main(string[] args)
53         {
54             Student s = new Student() { BirthDate = DateTime.Now.AddYears
55             (-9) };
56         }
57     }
58 }
```

```

55         Student.PrintMinAge();
56         Student.MinAge = 18;
57         // Program p=new Program();
58         // p.funcWithOptionalParameter(5);
59         funcWithOptionalParameter(i:5, y: 9,x:33);//named parameters
60         //Console.WriteLine("היא מחלקת המכילה פונקציות סטטיות לטיפול במסך");
61         //readline מחזירה מהרוצחת
62         //read מספר שהוא הערך האסורי של התו שהוקץ
63
64         var u = Console.Read ();
65         SumWithoutParams(new int[] { 4, 5, 6, 7, 8 });
66         // SumWithoutParams(7, 8, 9, 5); שגיאה
67         SumWithParams(3,6,7,8);
68         // לפונקציהcout כוכבשולחים ערך
69         // חיבבים לשלהו אותו בתור משנתנה
70         // bool y = func(10); שגיאה--
71         int x = 1000;
72         bool y = func(out x);
73         // ביתן להשתמש בערך של המשתנה שנשלח והשתנה במהלך הפעוקציה
74
75         int t =2;
76         funcWithRef(ref t);
77         t++;
78
79         if (x==7000)// yes
80         {
81             Console.WriteLine("נא הקלpkim");
82             string strNum = Console.ReadLine();
83             int number;
84             if (int.TryParse(strNum, out number)) // המרה בתנאי
85                 number++ ;
86
87             DateTime dt = new DateTime(2018, 7, 1);
88             DateTime dt2 = DateTime.Now;
89             DateTime t2= dt.AddMonths(2);
90             int i;
91             DateTime dt3;
92
93             Student dina = new Student();
94             Student p1 = new Student();
95             p1.Mother = dina;
96
97             Student p = new Student();
98
99             p.Grade1 = new Grade();
100            // p.Grade.Value = 100;// שגיאה. כי עדין לא נוצר מופע מהמחלקה
101            // צייר
102            p.Grade1.Value = 100;
103
104            p.GetFullName();
105        }
106    }
107

```

ט) הבדלים בין class - struct :

1. משתנה מסוג class יהיה תמיד RT [נוצר ב- heap], ומשתנה מסוג struct יהיה תמיד VT [נוצר ב- stack]
2. לא חייבים לכתוב לבנייה בנאי
3. לא חייבים לקרוא לבניאי – מכיוון שהוא נוצר מיד עם ההגדלה שלו. מילא, ניתן לגשת לרכיבים של משתנה מסוג struct שלא אוחח במנורש.
4. מבנה לא יכול להיות NULL
5. אין הורשה לבנייה או מבנה
6. לא ניתן לכתוב לו פונקציה הורשת

בחן על שיעור שני – 15 מתרוך 19 [כולל תשובות]

1. משתנה מסוג struct מוגדר ב- **V stack**
 2. משתנה מסוג struct שאינו מאותחל מכיל null **X**
 3. לכל מחלקה קיימים בניין גם אם לא כתבת אותו. **V**
 4. אין אפשרות לשלוח מבניין שמקבל פרמטרים לבניין שאינו מקבל. **X**
 5. בתכנית **net**. ניתן לשלב קוד מכמה שפות. **V**
 6. משתנה מסוג valueTypeNULL מכיל NULL אם לא אוטחל . **X**
 7. תכנית בשפה עילית ב-.NET עברת הידור רק במחשב הלוקה. **X**
 8. כבירית מחדל, חברי מחלקה יוגדרו כפרטים **V**
 9. **Person p= new Person();**
 - a. הצורה על משתנה –מצביע ב- **stack**
 - b. הקזאה של משתנה ב- **heap**
 - c. 2 התשובות שלעיל אין נוכנות
 - d. שתי התשובות הראשונות נכונות **V**
10. פרמטר מסוג params יכול לקבל רשימת ערכים מכל סוג שהוא:

X Public void p(params int[] parArr)

11. ניתן לשנות את ערכו של משתנה const רק במבנה **X**
12. ניתן לשנות את ערכו של משתנה סטטי בכל שלב בתכנית. **V**
13. גישה למשנה סטטי נעשית באמצעות מופע. **X**
14. לא ניתן לגשת למשנה המופיע בתוך פונקציה סטטית. **V**
15. מאפיינים הם למעשה פונקציות. **V**
16. לכל מאפיין חייב להיות ערך פרטוי התואם לשם אך באות קטינה. **X**
17. יש ליצור מופע מהמחלקה **Console** כדי להשתמש בה. **X**
18. הגדרה של enum היא הגדרת טיפוס נתונים חדש. **V**
19. כדי להשתמש ב- enum יש ליצור ממנו מופע. **V**

בחן על שיעור שלישי - מערכים ואוסףים, העמסת אופרטורים [25 מתוך 33, בחירת רשות מראש בלבד]

סמי נכון או לא נכון:

1. מערך של int הוא מטיפוס value type ✓
2. לא ניתן לזרז שחרור של זיכרון שנתפס על ידי מערך. ✓
3. את ה- Garbage Collector בדרך כלל מפעילים על פי בקשה. ✓
4. עיני בקוד הבא: ✓

Person p1=new Person(){Name="Chaim"};

Person p2=new Person(){Name="Chaim"};

Var b=p1==p2;

במשתנה b יהיה ערך:

True •

False •

• המשפט יתן שגיאה

5. מערך מהיר יותר מאשר אוסף. ✓
6. משתנה מערך שוקולדים הוא reference type ✓
7. כל איבר במערך הנ"ל הוא value type ✓
8. כאשר רשימה רכיבי השוקולד [Ingredients] הינה מערך מחוזות . המאפייןIngredients הינו RT ✓
9. כאשר רשימת רכיבי השוקולד הינה מערך אובייקטים - המאפיין ChocolateIngredients הינו VT ✓
10. לפי תשובה קודמת – האם כל אחד מרכיבי השוקולד מצריך now או לא? ✓
11. כאשר נקצת 2 חבילות נופש המצביעות לאותו מלון. יוצרו 2 אובייקטים מסוג מלון. ✓
12. כאשר נוסיף שוקולד לרשימה השוקולדים שבמערך – בהכרח שהוא יכול גם את אותה רשימת רכיבים. ✓
13. כאשר נשנה איבר ברשימה הרכיבים המשותפת – לכל גDAL השוקולד-פרווה. נוצרה לשנות אותו בנפרד גם לכל אחד מ4 האובייקטים שוקולד-פרווה? ✓
14. String הוא reference type אשר מתנהג כמו Value type ✓
15. במערך זו ממדי מספר השורות ומספר העמודות שווה. ✓
16. Jagged array יכול להכיל מספר בלתי ידוע מראש של מערכים. ✓
17. הפעולה: X

int[][] source= new int[10][];

int[][] copy = source;

מעתיקת את ערכי המערך בעותק חדש. ✓

18. למערך יש פונקציות מוכנות שהוא יורש ממחלקת `ArrayList`.

19. לא ניתן לגשת לערך במערך ממין באמצעות אינדקס.

20. לא ניתן לשולח פרמטרים לפונקציה `main` מכיוון שהיא אינה נקראת על ידי המתכנת.

21. `ArrayList` היא מחלוקת הטרוגנית, ויכולת להכיל סוגי מגוונים של טיפוסים.

22. במחלוקת `Hashtable` כל אחד מאברי האוסף הוא מסוג `DictionaryEntry`.

23. ניתן לגשת לערכים בראשימה זו באמצעות האינדקס שלהם או באמצעות המפתח.

24. המפתח אינו יכול להיות `NULL`, הערך יכול להיות `NULL`.

25. מספר הערכים באוסף מסוג `Hashtable` אינו מוגבל.

26. המפתח והערך חייבים להיות מאו טיפוס נתוניים.

27. לאבירי המחלוקת `sortedList` ניתן לגשת גם באמצעות אינדקס.

28. המחלוקת `Queue` הינה מסוג `LIFO` - `FIFO` - `Circular Queue`.

29. כאשר שולפים איבר מהחלוקת `Stack` באמצעות הפונקציה `pop` – הערך מסיר מהראשימה.

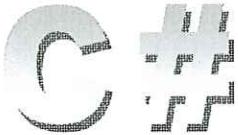
30. ניתן לבצע `Person+Person` באמצעות העמסת אופרטורים.

31. פונקציה המעבילה אופרטור אינה יכולה להיות סטטית.

32. אופרטור בינהרי הינו אופרטור הפועל על שני ארגומנטים.

33. אופרטור אונארי `++` חייב להחזיר ערך מסווג המצביע עליה ממשים את האופרטור.

בצלחה !



בחן על שיעור 4:

1. בשפת # C++, מכמה מחלקות ניתן לרשת כדי ליצור מחלקה חדשה ייחודה?

2. מה ההבדל בין `private` ו- `protected` ?

`private` - מוגבל לתוכה בלבד, `protected` - מוגבל לתוכה ותת-מחלקות.

3. כיצד מוסתרת שיטה במחלקה בסיסי?

4. כיצד ניתן להבטיח שכל מחלקה נגזרת תיצור גרסה משלה לשיטה?

public abstract class Base { ... } public class Derived : Base { ... }

5. איזו מילה מפתח משמשת למנוע הורשה ממחלקה?

`sealed`

6. איזו מילה מפתח משמשת למנוע יצירת אובייקטים ממחלקה?

`static` - מוגבל לתוכה בלבד, `private` - מוגבל לתוכה בלבד, `abstract` - מוגבל לתוכה בלבד.

7. מהי מחלקה הבסיסי היסודית מנתה כל המחלקות נגזרות?

`object`

8. צייני 3 שיטות שכל האובייקטים מכילים אותם.

`HashCode`, `GetType`, `Equals`

9. מהי אריזה? ומהי פריקה?

`stack`, `heap`, `store`, `queue`, `list`, `array`, `linked list`, `hash map`, `dictionary`, `set`, `tuple`.

10. למה משמשת מילה מפתח AS?

מה ההבדל בין לבן? IS

11. מתי קיבלתי הודעה שגיאה זו:

cannot declare a body because it is marked abstract

does not implement inherited abstract member 'SuperMarket.product.productName.get'

12. מהי המריה כלפי מטה?

13. מהי המריה כלפי מעלה?

14. מהי המריה כלפי מטה?

15. מהי המריה מרומצת ומתי היא אפשרית?

16. מהי המריה מפורשת? מתי היא מחייבת וכיצד נעשה זאת (2 דרכים)

17. מה צריך לעשות כדי שנוכל להתייחס למאפיין "הקשר" שאינו ברשימה:

```

product f111=new Food("bread",3.99,ECHSHER);
Food f222=new Food("lightBread",8.99,ECHSHER);
f222=f111.*;
product parray={ Equals, product(10),
GetHashCode, 22;r33;r44;r55;f111;f222,
foreach pId, pArray
{
    productName
    productPrice
    ToString
}
Warning 0
  
```