יסודות מדעי המחשב 1

מדריך מעבדה לסביבת העבודה

Visual C# Express

כתבה: יעל בילצ׳יק (סופרין)

מהדורת עיצוב

תשס״ו 2006

אוניברסיטת תל-אביב החוג להוראת המדעים

מטה מל"מ המרכז הישראלי להוראת המדעים ע"ש עמוס דה-שליט

משרד החינוך האגף לתכנון ולפיתוח תכניות לימודים

אין לשכפל, להעתיק, לצלם, לתרגם או לאחסן במאגר מידע כל חלק שהוא מחומר הלימוד של ספר זה. שימוש מסחרי מכל סוג שהוא בחומר הכלול בספר זה אסור בהחלט, אלא ברשות מפורשת בכתב מהגורמים המפורטים להלן.

©
כל הזכויות שמורות

אוניברסיטת תל-אביב ומשרד החינוך

תוכן העניינים

פתיחת סביבת העבודה..... 5

יצירת פרויקט..... 5

חלונות סביבת העבודה..... 6

כתיבת תוכנית ראשונה..... 7

הידור תוכנית..... 8

הרצת תוכנית..... 9

שמירת תוכנית..... 10

יצירת פרויקט נוסף..... 11

הקלדת קלט..... 11

ניפוי שגיאות..... 12

עבודה מתקדמת עם מנפה השגיאות..... 15

טבלת מקשי קיצור שימושיים לפעולות בסביבת העבודה:..... 16

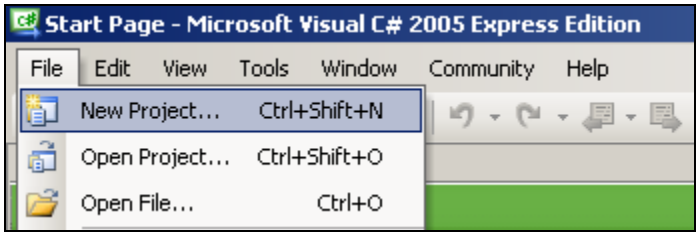
סביבת העבודה Visual C# Express מאפשרת לנו לכתוב תוכניות בשפת #C, החל מתוכניות פשוטות למדי בעלות שורות קוד בודדות, ועד מערכות מסחריות מורכבות בעלות אלפי שורות קוד. במדריך זה נלמד כיצד להשתמש בסביבה זו באופן הנוח ביותר לצרכינו כמתכנתים מתחילים.

פתיחת סביבת העבודה

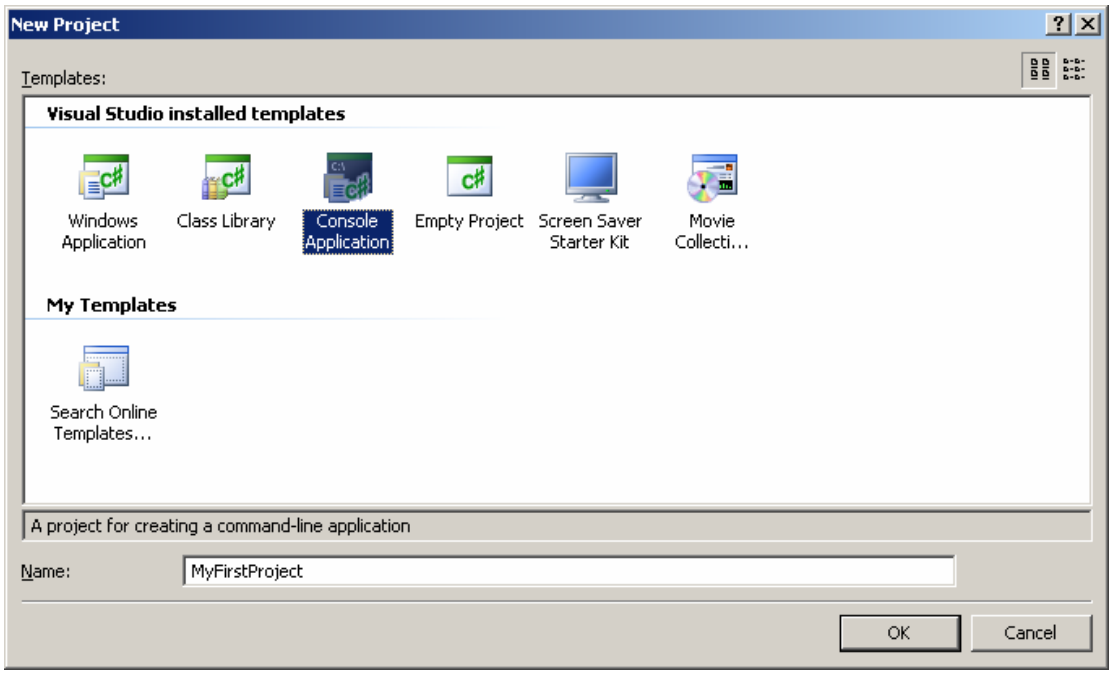
כדי לפתוח את סביבת העבודה נבחר את התוכנה Microsoft Visual C# 2005 Express Edition, דרך תפריט ההתחלה של "windows". יתקבל מסך הפתיחה של סביבת העבודה.

יצירת פרויקט

כל תוכנית בשפת #C נמצאת בתוך פרויקט נפרד. לכן, כדי לכתוב תוכנית חדשה עלינו לפתוח פרויקט חדש. נבחר בתפריט העליון את File, ושם נבחר ב-New Project.



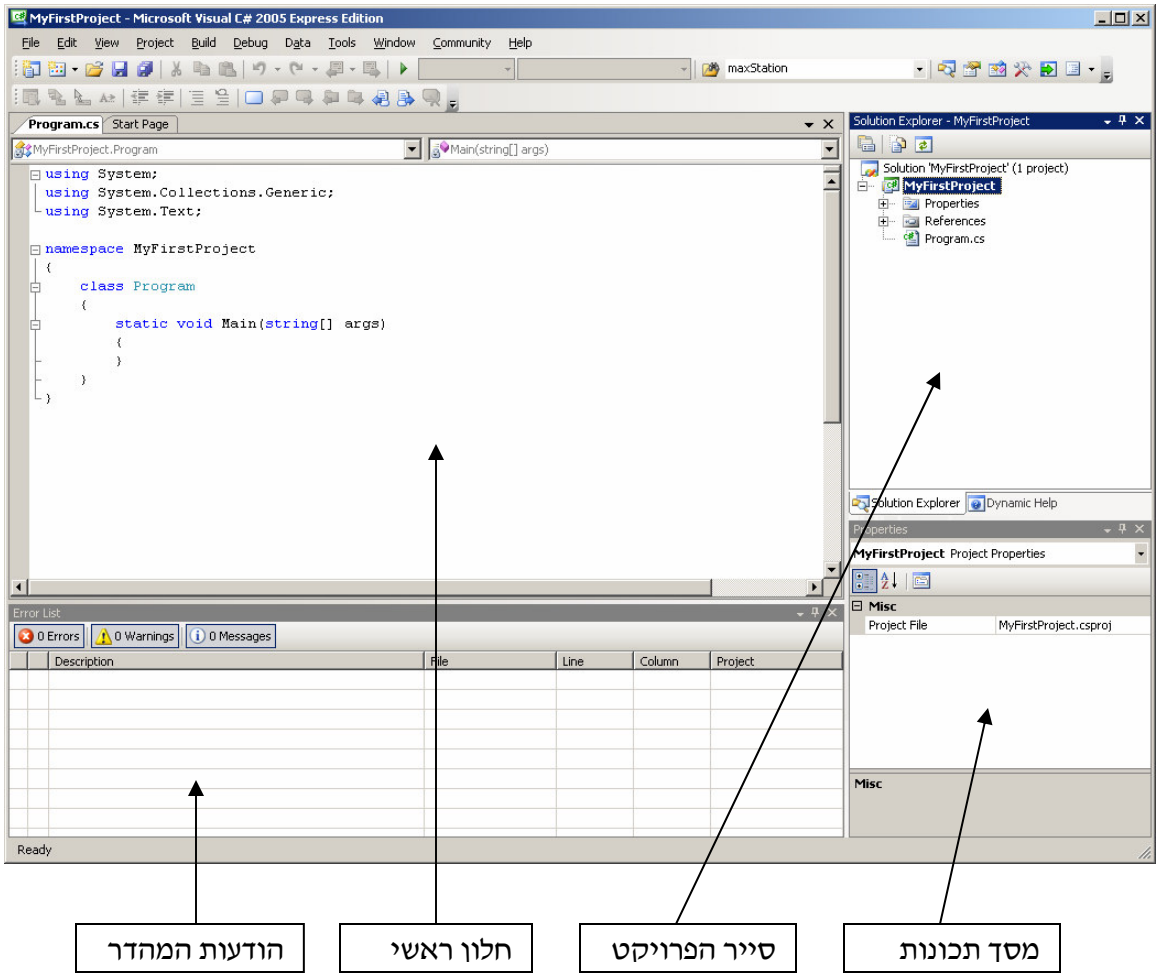
ייפתח חלון המציע לנו סוגי פרויקטים שונים. נבחר ב-Console Application. בשורת השם נכתוב את השם שנקרא לפרויקט. בדוגמה זו בחרנו בשם "MyFirstProject"



לאחר הלחיצה על "OK" יתקבל מסך סביבת העבודה, כאשר בחלון המרכזי כבר מופיע בסיס של תוכנית ריקה בשפת #C.

חלונות סביבת העבודה

נכיר את החלונות השונים בסביבת העבודה:



החלון הראשי ישמש אותנו לכתיבת קוד התוכנית

בחלון **הודעות המהדר** נצפה בשגיאות והערות המהדר (קומפיילר) לאחר כל הידור (קומפילציה).

בחלון **סייר הפרויקט** נשתמש על מנת לעבור בנוחות בין חלקי הפרויקט השונים. ניתן לראות כי הפרויקט שפתחנו קיים בתוך Solution. כאשר פתחנו פרויקט חדש נפתח עבורו Solution חדש. משמעות המילה solution היא פתרון. בסביבת העבודה, Solution שומר בתוכו פרויקט אחד או יותר, ומקובל שיהיו אלה פרויקטים שיש ביניהם קשר, והם מהווים במובן מסוים פתרון לבעיה מורכבת אחת. בהמשך נלמד כיצד ניתן ליצור כמה פרויקטים בתוך Solution אחד.

חלון **מסך התכונות** משמש בעיקר עבור תוכניות בהן נעשה שימוש בעזרים גראפיים ולכן לא נשתמש בו.

ניתן לסגור כל חלון אם אין בו שימוש, ולפתוח אותו שוב בעזרת התפריט View.

נתבונן בקוד (כלומר, רצף הוראות בשפת התכנות) אשר מופיע באופן אוטומטי במסך התוכנית עם פתיחת פרויקט חדש:

```
using System;
using System.Collections.Generic;
using System.Text;

namespace MyFirstProject
{
    class Program
    {
        static void Main(string[] args)
        {
        }
    }
}
```

רשימת ספריות

הגדרת המרחב

הגדרת המחלקה

הגדרת נקודת הפתיחה לתוכנית

לא כל ההגדרות הכרחיות לצרכינו, לכן נוכל למחוק את ההגדרות שלא נזדקק להן ולהישאר עם השלד המוכר לנו:

```
using System;

class Program
{
    static void Main()
    {
    }
}
```

כתיבת תוכנית ראשונה

כעת אנו יכולים להקליד את התוכנית הראשונה, שתציג למסך את הפלט "Hello world".

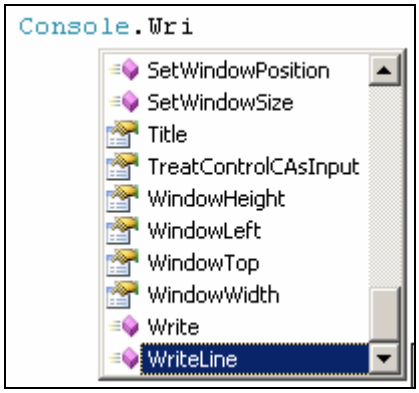
נשנה את שם המחלקה לשם שנרצה לקרוא לתוכנית (אין חובה לשנות את השם, אך רצוי תמיד לתת לתוכניות שמות משמעותיים, שמביעים את תפקידן), ונוסיף את פקודת ההדפסה בתוך תחום ה-Main:

```
using System;

class HelloWorld
{
    static void Main()
    {
        Console.WriteLine("Hello world");
    }
}
```

שימו ♥:

- ישנם צבעים שונים בקוד הכתוב: מילים שמורות נכתבות בכחול, מחרוזות באדום, שמות מחלקות בטורקיז.
- מייד לאחר כתיבת שם המחלקה Console נפתח חלון המציג את כל התכונות והפעולות של המחלקה בהן נוכל להשתמש. לפקודת הדפסה נבחר את הפעולה WriteLine.



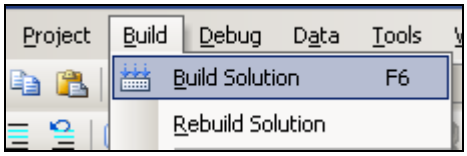
תוכלו לעבור ולבדוק תכונות ופעולות נוספות השייכות למחלקה Console.

למשל, ההוראה `Console.ForegroundColor = ConsoleColor.Red;` תשנה את צבע הפלט לאדום.

- כדי לקבל הנחיות על כל תכונה ופעולה אפשר לעמוד עם הסמן על התכונה או הפעולה המבוקשת ולהקיש על F1.

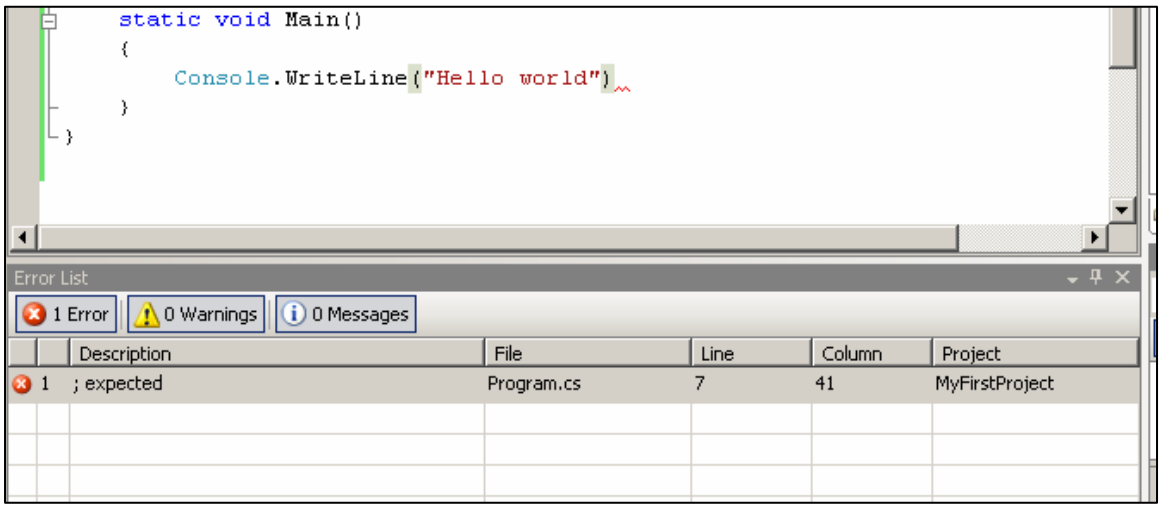
הידור תוכנית

אחרי סיום הקלדת התוכנית, עלינו להדר (לקמפל) אותה כדי לבדוק שאין בה שגיאות תחביריות, ולהכין אותה לריצה. שלב זה יתבצע על ידי בחירת Build בתפריט, ואז בחירת Build Solution, או על ידי הקשה על המקש F6.



אם אין כלל שגיאות תחביר בתוכנית, נקבל את ההודעה Build succeeded בצד השמאלי התחתון של המסך. אם קיימות שגיאות תחביר בתוכנית נקבל הודעות מתאימות בחלון הודעות המהדר. לחיצה כפולה על ההודעה תקפיץ את הסמן למקום בו ארעה השגיאה.

למשל, השגיאה הבאה נגרמה כיוון שלא נכתב הסימן ; בסוף משפט:



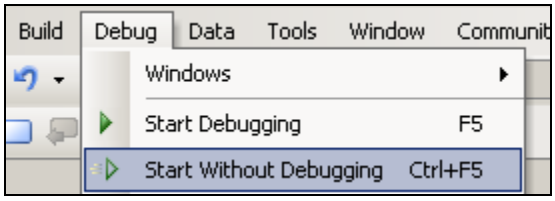
לא נוכל להריץ תוכנית לפני שנתקן את כל שגיאות התחביר. לאחר שנתקן את השגיאות, נהדר שוב את התוכנית, וכך נחזור על התהליך עד אשר לא יופיעו שגיאות בחלון הודעות המהדר, ונקבל את ההודעה Build succeeded.

שימו ♥: ייתכן שלאחר הידור של תוכנית נקבל **הערות** הרצה. ההערות מסומנות בסימן צהוב (בעוד השגיאות מסומנות באדום). ניתן להריץ תוכנית שהתקבלו עבורה הערות, אך יש לתת את הדעת על הערות אלה, משום שייתכן שהן מצביעות על טעות או על בעיה אפשרית אחרת בתוכנית.

הרצת תוכנית

לאחר שהתוכנית עברה בהצלחה את שלב ההידור, היא מוכנה להרצה.

כדי להריץ את התוכנית נבחר בתפריט Debug את Start Without Debugging או נקיש על המקשים Ctrl ו-F5 בו זמנית.



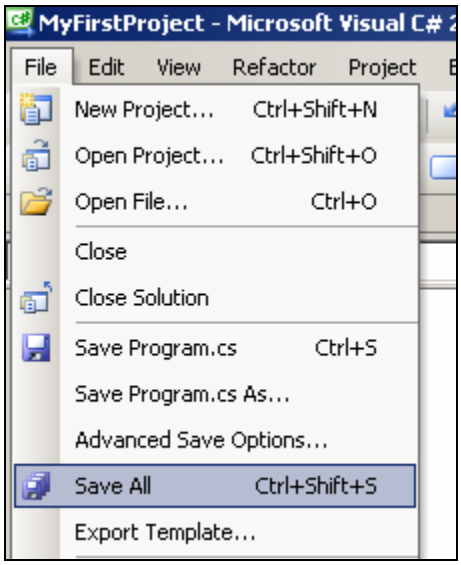
כתוצאה מכך, יפתח חלון ריצת התוכנית, ייכתב בשורה נפרדת המשפט "Hello world", וריצת התוכנית תסתיים. בסיום ריצת התוכנית חלון ההרצה ייסגר רק לאחר הקשה על מקש כלשהו.

```
Hello world
Press any key to continue . . .
```

שימו ♥: אם תנסו להריץ תוכנית אשר לא עברה הידור בהצלחה, או תוכנית שעברה הידור אך לאחר מכן עברה שינוי כלשהו ויש להדרה שוב, יתבצע הידור באופן אוטומטי.

שמירת תוכנית

לאחר שסיימנו לכתוב את התוכנית, להדר אותה (תוך תיקון שגיאות תחביר, במידת הצורך), להריץ אותה, ולבדוק שאין שגיאות לוגיות בתוכנית, נשמור את קובץ התוכנית לשימוש עתידי. בתפריט File בחרו את Save All או הקישו על המקשים Ctrl, Shift ו-S בו זמנית.



ייפתח חלון בו יופיע שם הפרויקט, הכתובת על גבי הדיסק הקשיח בה יישמר הפרויקט, ושם ה-Solution.

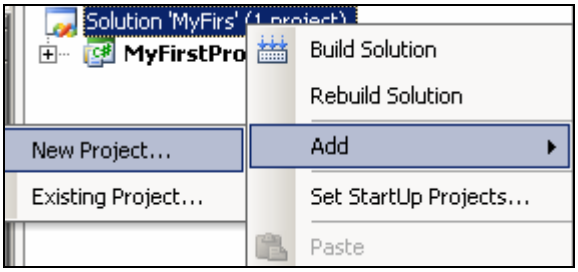
כאשר נרצה בעתיד לפתוח תוכנית שמורה נפתח את הפרויקט שלה על ידי Open project שבתפריט File, ונבחר את ה-Solution הנדרש לפי שמו בסיומת sln.

יצירת פרויקט נוסף

כעת ברצוננו לכתוב תוכנית נוספת, הדורשת גם הזנת קלט.

לפנינו שתי אפשרויות: לסגור את ה-Solution הקיים על ידי Close Solution שבתפריט File, ולפתוח פרויקט חדש כפי שעשינו בתחילה.

אפשרות שנייה היא להוסיף פרויקט ל-Solution קיים. לשם כך ניגש עם העכבר למילה Solution שבסייר הפרויקט, נלחץ על המקש הימני של העכבר ונבחר ב Add. מהתפריט שייפתח נבחר את NewProject.



כעת ייפתח לנו חלון פרויקט חדש, כפי שקרה כאשר פתחנו Solution חדש.

שימו לב שלאחר הוספת הפרויקט קיימים שני פרויקטים תחת אותו ה-Solution, אך רק אחד מהם פעיל – זה המסומן בהדגשה. ניתן להחליף ולסמן פרויקט שונה כפעיל על ידי בחירתו עם העכבר, הקשה על מקש ימני ובחירת Set as StartUp Project.

כאשר אנו מוסיפים פרויקט ל-Solution, המכיל כבר פרויקטים אחרים, הרי שבסופו של דבר הפרויקטים יישמרו יחדיו, תחת אותו Solution (ואיתם גם התוכנית שכתבתם בכל אחד מהם). לכן, אם אתם כותבים כמה תוכניות בעלות אופי דומה שברצונכם לשמור יחדיו (למשל, כמו כמה תרגילים עבור עבודה אחת להגשה באותו נושא) מומלץ לשמור אותן בפרויקטים נפרדים תחת אותו ה-Solution. אך אם הינכם כותבים כמה תוכניות שאין כל קשר ביניהן, מומלץ לשמור כל אחת מהן ב-Solution נפרד עם שם משמעותי שיקל עליכם את הזיהוי של התוכנית כאשר תחפשו אותה בקבצים השמורים.

הקלדת קלט

כידוע, זהבה גרמה נזק רב לשלושת הדובים, ולכן החליטה לפצותם בסכום כסף. את הסכום תחלק שווה בשווה בין כל השלושה. הדוב הקטן החליט שאת סכום הכסף שקיבל יחלק לארבעה חסכונות נפרדים, ואת השארית יבזבז על צנצנת דבש איכותית. עלינו לכתוב תוכנית שתקבל כקלט את סכום הכסף שהקצתה זהבה לתשלום הפיצויים, ותציג כפלט את סכום הכסף שיוכל הדוב הקטן לבזבז על צנצנת דבש איכותית.

הנה תוכנית שנכתבה לצורך פתרון הבעיה, כפי שהוקלדה בסביבת העבודה:

```
using System;

class Bears
{
    static void Main()
    {
        int money, smallBearSum, sumForHoney;
        Console.Write("Insert the sum of money Zeahva has: ");
        money = int.Parse(Console.ReadLine());
        smallBearSum = money / 3;
        sumForHoney = smallBearSum / 4;
        Console.WriteLine("The bear junior will spend {0} shekels for
honey", sumForHoney);
    }
}
```

הקלידו גם אתם את התוכנית הזאת, והדרו אותה.

לאחר שהתוכנית עברה בהצלחה את תהליך ההידור, נריץ אותה.

בתחילה יירשם במסך ריצת התוכנית משפט הפלט

```
"Insert the sum of money Zeahva has: "
```

ולאחר מכן התוכנית תעצור פעולתה ותמתין לקלט של מספר שלם. עלינו להקליד בהמשך למשפט זה מספר שלם כלשהו ואחריו להקיש על המקש Enter. רק אז התוכנית תמשיך את ריצתה, תציג את הפלט המחושב ותסיים את פעולתה.

שימו ♥: אם התוכנית ממתינה לקבל מספר שלם, אך אנו נקליד קלט שאינו מספר שלם, יגרום הדבר לשגיאת ריצה. ייפתח מסך בו תופיע השאלה: האם ברצונכם לנפות את השגיאות באמצעות כלי לניפוי שגיאות? לחצו על "No". אז תופיע הודעת השגיאה על גבי מסך התוכנית, והתוכנית תפסיק ריצתה.

ניפוי שגיאות

נתבונן במסך ריצת התוכנית לאחר ריצתה, כאשר הקלט הוא 100:

```
Insert the sum of money Zehava has: 100
The bear junior will spend 8 shekels for honey
Press any key to continue . . .
```

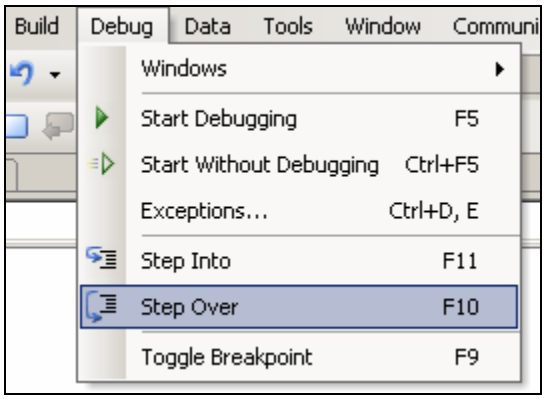
נבחן את התוצאה שהתקבלה:

אם זהבה הקצתה 100 ₪ לפיצויים, הרי כל דוב יקבל 33 ₪. הדוב הצעיר יחלק את הסכום שקיבל, 33 ₪, ל-4 קבוצות ובשארית שיקבל יקנה צנצנת דבש איכותית. שארית החילוק של 33 ב-4 היא 1, לכן הפלט צריך להיות 1. אם נתבונן במסך הפלט נראה כי הפלט הוא 8. מכאן, שיש בתוכנית שכתבנו שגיאה לוגית, שגיאת חישוב במהלך התוכנית.

בתוכנית קצרה כגון זו שכתבנו ניתן להתבונן בתוכנית ולמצוא את השגיאה בקלות יחסית. אך כאשר התוכנית גדולה ומורכבת יותר ניפוי השגיאות הופך למשימה קשה הרבה יותר. לשם כך קיים כלי המאפשר לנו לנפות שגיאות ביתר קלות, ה-Debugger.

בעזרת ה-Debugger ניתן להריץ את התוכנית באופן מבוקר, שורה אחר שורה, בכל שורה נצפה בערך המשתנים ונבדוק כי ערכם תואם לערך הצפוי.

לתחילת הרצה מבוקרת של התוכנית בחרו מהתפריט Debug את Step Over או הקישו על F10.



ריצת התוכנית תחל והשורה הראשונה של התוכנית תיצבע בצהוב. כעת, כל הקשה על F10 תגרום להתקדמות התוכנית לשורה הבאה. לחצו על F10 עד אשר תתבצע שורת הפלט הראשונה וראו כי אכן נרשמה על מסך ריצת התוכנית שורת הפלט. לחיצה נוספת על F10 תגרום להמשך ריצת התוכנית אל השורה הבאה, שורת הקלט. מכאן תוכלו להמשיך את ריצת התוכנית רק לאחר שתזינו את הקלט המבוקש. הקלידו את המספר 100 במסך ריצת התוכנית והקישו Enter. כעת חזרה השליטה לתוכנית.

נתבונן במסך "Locals" הנפתח בצד שמאלי תחתון של המסך:

במסך "Locals" נוכל לצפות בערכי המשתנים בעת ריצת התוכנית. שימו לב לרגע המתואר בצילום: על פי השורה הצבועה בצהוב ניתן לדעת כי כבר נקרא הקלט 100 ממסך ריצת התוכנית, והושם במשתנה money. נתבונן במשתנה זה במסך "Locals" ונבחין כי אכן המשתנה קיבל את הערך 100. הערך צבוע בצבע אדום מכיוון ששורת התוכנית האחרונה שהתבצעה גרמה לשינוי ערך המשתנה.

```
class Bears
{
    static void Main()
    {
        int money, smallBearSum, sumForHoney;
        Console.Write("Insert the sum of money Zehava has: 
        money = int.Parse(Console.ReadLine());
        smallBearSum = money / 3;
        sumForHoney = smallBearSum / 4;
        Console.WriteLine("The bear junior will spend {0} s
    }
}
```

Locals

Name	Value	Type
money	100	int
smallBearSum	0	int
sumForHoney	0	int

Call Stack

Name
Bears.exe!Bears.M
[External Code]

השורה הבאה בתוכנית אמורה לחשב את ערך המשתנה smallBearSum. לפני שנמשיך את ריצת התוכנית נחשוב מה ערכו של משתנה זה אמור להיות. כפי שחישבנו קודם, כל דוב יקבל 33 ₪ ולכן זה צריך להיות ערכו החדש של המשתנה. נמשיך את ריצת התוכנית על ידי הקשה נוספת על המקש F10. התבוננו במסך "Locals" וראו כי המשתנה smallBearSum שינה כעת את ערכו מ-0 ל-33, כצפוי. אם כך, עד עתה התוכנית עבדה באופן תקין.

השורה הבאה אמורה לחשב את ערך המשתנה sumForHoney. שוב, נחשוב מה אמור להיות ערכו של משתנה זה. כפי שחישבנו קודם, שארית החלוקה של 33 ב-4 היא 1, ולכן זו התוצאה המבוקשת. הקשה נוספת על המקש F10 תמשיך את ריצת התוכנית לשורה הבאה, וערכו של המשתנה sumForHoney משתנה לערך 8, ולא לערך הצפוי. אם כך, בשורה זו ישנה שגיאה.

נתבונן בשורת הקוד הבעייתית:

```
sumForHoney = smallBearSum / 4;
```

כוונתנו הייתה לחשב את **שארית החילוק** של `smallBearSum` ב-4, ובמקום זאת חישבנו את **תוצאת החילוק** של `smallBearSum` ב-4. שגיאה זו אירעה כיוון שהשתמשנו בסימן / במקום בסימן %.

נתקן את השגיאה בקוד התוכנית כך שהשורה תיראה כעת כך:

```
sumForHoney = smallBearSum % 4;
```

לאחר שתיקנו את השגיאה, נרצה לבדוק האם כעת התוכנית נכונה ומציגה את הפלט הנכון.

נוכל לבצע זאת באחת משתי הדרכים הבאות:

1. נעצור את ריצת התוכנית (על ידי Stop Debugging שבתפריט Debug, או על ידי הקשה על המקשים shift ו-F5 בו זמנית), ולאחר מכן נריץ את התוכנית שוב, לאחר השינוי, מההתחלה, על ידי ריצה רגילה של התוכנית, או על ידי ריצה תוך כדי ניפוי שגיאות.
2. נגרום לשורה שתיקנו להתבצע שוב: משמאל לשורות הקוד הצבוע בצהוב (השורה הבאה לביצוע) נבחין בחץ צהוב. נגרור את החץ הצהוב חזרה לשורת הקוד המתוקנת על ידי העכבר, כך ששורה זו תתבצע שוב מחדש, ונמשיך את ריצת התוכנית על ידי המקש F10 עד לסיומה.

נחזור על תהליך ניפוי השגיאות עבור כל שורה בתוכנית, ונבצע את המעקב עד אשר נהיה בטוחים כי התוכנית מספקת פלט נכון עבור כל קלט אפשרי.

שימו ♥: בשלב זה שימוש במקש F11 יהיה זהה עבורנו לשימוש במקש F10. בהמשך לימודינו, כאשר נרצה להיכנס ולבדוק פעולות בקוד שנכתבו על ידינו, נשתמש במקש F11, ואילו מעבר על פניהן מבלי להיכנס ולבדוק אותן יבוצע על ידי המקש F10.

עבודה מתקדמת עם מנפה השגיאות

בהמשך לימודינו נכתוב תוכנית ארוכות, ולא נרצה לעבור על כל שורות הקוד בחיפושינו אחר שגיאה. לכן, אפשרות נוספת לעבודה עם ה-Debugger היא על ידי הרצת התוכנית באופן רציף, עד נקודה מסוימת, בה תעצור התוכנית את ריצתה. לשם כך נשתמש בנקודות עצירה, ה-Breakpoint.

למשל, נניח שאנו משוכנעים כי בתוכנית Bears קליטת נתון הקלט נעשתה כיאות ואין צורך לבדוק זאת. אם כך, אנו יכולים להציב נקודת עצירה בשורה שאחרי קליטת הקלט. לשם הצבת נקודת עצירה בשורה מסוימת נעמוד עם הסמן על השורה המבוקשת, ונבחר את Toggle Breakpoint שבתפריט Debug, או שנקיש על המקש F9. דרך נוספת להצבת נקודת עצירה היא על ידי לחיצה על המקש השמאלי של העכבר בשטח האפור שמשמאל לשורת הקוד המבוקשת.

כתוצאה מכך תופיע נקודה אדומה משמאל לשורת הקוד המבוקשת, והשורה תיצבע באדום.

```
int money, smallBearSum, sumForHoney;
Console.Write("Insert the sum of money
money = int.Parse(Console.ReadLine());
smallBearSum = money / 3;
sumForHoney = smallBearSum / 4;
```

כעת נריץ את התוכנית, אך לא באופן הרגיל, אלא במצב ניפוי שגיאות, על ידי בחירת Start Debugging בתפריט Debug, או על ידי הקשה על המקש F5.

התוכנית תתחיל את ריצתה, תדפיס את שורת הפלט, תעצור לבקש קלט, נקליד 100 ו-Enter, וכעת, כאשר התוכנית תגיע לשורה המסומנת על ידי נקודת-עצירה, תפסיק את ריצתה ותעצור במצב Debugging המוכר לנו. ממצב זה, כפי שכבר ראינו, ניתן להמשיך להריץ שורה אחר שורה על ידי

המקש F10. אפשרות נוספת היא להוסיף נקודות-עצירה נוספות לאורך התוכנית ולרוץ מאחת לשנייה על ידי המקש F5. בכל הקשה על F5 התוכנית תרוץ מהמקום האחרון בו היא עצרה ועד נקודת-העצירה הבאה. אם לא תמצא נקודת-עצירה נוספת, תרוץ התוכנית עד לסיומה.

טבלת מקשי קיצור שימושיים לפעולות בסביבת העבודה:

פעולה	הסבר	מקש קיצור
Build Solution	הידור התוכנית	F6
Start Without Debugging	הרצת התוכנית	Ctrl + F5
Start Debugging	הרצת תוכנית עד נקודת עצירה	F5
Step Over	קידום הרצת התוכנית בשורה אחת מבלי להיכנס לפעולות בקוד	F10
Step Into	קידום הרצת התוכנית בשורה אחת כך שניכנס לפעולות בקוד שנכתבו על ידינו	F11
Toggle Breakpoint	הצבה או הסרה של נקודת עצירה על השורה בה נמצא הסמן	F9 או מקש שמאלי של העכבר
Format Document	הזחת התוכנית (אינדנטציה)	Ctrl+E, D