Division & Modulo

חילוק ושארית

עד עתה, הפעולות החשבוניות-אופרטורים שהכרנו בתכנות פעלו כפי שהכרנו אותן ממתמטיקה, כשנסתכל על פעולת החילוק, נראה שהיא מתנהגת מעט אחרת.

תזכורת: מספר שלם-int מספר עשרוני-double

פעולת החילוק תתנהג אחרת רק כשמדובר בחילוק מספר שלם למספר שלם.

לדו', 8/3, 9/4 וכו'. אותה התנהגות תתקיים גם כאשר החילוק הוא בין משתנים מטיפוס int.

לעומת זאת, אם החילוק בין עשרוני לשלם, בין double ל int הפעולה תפעל כמו במתמטיקה.

<u>מתמטיקה לעומת תכנות:</u>

1. חלוקה מתמטית: במתמטיקה, כאשר מחלקים שני מספרים שלמים, התוצאה תמיד תהיה מדויקת, עשרונית או שלימה בהתאם לפתרון התרגיל. לדו' 7/3=2.3333
2. חילוק שלמים בתכנות: ב C#, כמו ברוב שפות התכנות חלוקת שני מספרים שלמים עם האופרטור `/` מחזירה רק את חלק המספר השלם של החלוקה, תוך ביטול החלק השברי. לדו', 7/3=2 ולא 2.3333 הסיבה לכך היא שגם `7` וגם `3` הם מספרים שלמים, וחלוקת מספרים שלמים חותכת את החלק העשרוני.
3. פעולת מודולו (%): האופרטור modulo (%) ב-C# מחזיר את יתרת החלוקה בין שני מספרים שלמים. לדו', 7%3=1 מכיוון שכאשר 7 מחולק ב-3, היתרה היא 1 (מכיוון ש 7=3*2+1)

<u>המרה ל double כדי לקבל תוצאה מדויקת</u>

כדי לקבל תוצאה מדויקת בעת חלוקת שני מספרים שלמים ושמירה על החלק העשרוני, נוכל:

אפשר להמיר את אחד המספרים השלמים או את שניהם ל double לפני ביצוע החילוק. לדו':

```
int a = 7;
int b = 3 ;
double result = (double)a / b ;
```

זה מבטיח שהחלוקה תתבצע כחלוקה עם מספר עשרוני, תוך שמירה על החלק העשרוני.

<u>שימושים שונים:</u>

<u>בדיקה האם מספר הוא זוגי/אי זוגי:</u>

כשנשתמש באופרטור 2% (שארית) על כל מספר שהוא, אם המספר זוגי-התוצאה תהיה 0, אם המספר אי זוגי-התוצאה תהיה 1. לדו': 18%2=0, 87%2=1, 42%2=0, 7%2=1...

<u>פירוק מספר לספרות:</u>

לעיתים מספר הוא בעצם רצף של ספרות, שכל ספרה בו מייצגת משהו. ונרצה "לפרק" את המספר לספרות.

קבלת ספרה הימנית של מספר-ספרה ימנית תמיד תהיה ספרת האחדות, ולכן על כל מספר שהוא, לא משנה מכמה ספרות יהיה מורכב, נוכל להשתמש ב 10% כדי לקבל את ספרת האחדות.

לדו': ...**5**%10=**5**, 7**4**%10=**4**, 25**8**%10=**8**

קבלת ספרה שמאלית של מספר-כאן העניין נהיה מורכב מעט יותר. מכיוון שכל פעם הספרה השמאלית תייצג מיקום שונה, בהתאם להרכב המספר. לדו', במספר חד ספרתי הספרה השמאלית תהיה ספרת האחדות, במספר דו ספרתי זו תהיה ספרת העשרות, במספר תלת ספרתי ספרת המאות וכן הלאה. לבן, נשתמש כאן באופרטור החילוק /, נחלק בחזקת 10 המתאימה ע"פ מיקום הספרה השמאלית במספר אותו נרצה "לפרק". לדו':

במספר תלת ספרתי זו תהיה ספרת המאות, ולבן נחלק ב 100: 859/100=8

במספר דו ספרתי זו תהיה ספרת העשרות, ולבן נחלק ב 10: 41/10=4

וכו'.

קבלת ספרות אמצעיות של מספר-כדי לקבל ספרה אמצעית של מספר, נצטרך "להביא" אותה למצב בו היא תהיה הספרה הימנית ביותר או השמאלית ביותר. לשם כך נצטרך לבצע פעולה חשבונית שתקבל את המספר ללא ספרותיו מימין/שמאל. לדו', כדי לקבל את הספרה האמצעית ממספר תלת ספרתי, נצטרך לגרום לספרת העשרות להיות מימין או משמאל, ז"א "להוריד" את ספרת האחדות או המאות.

כיצד נעשה זאת?

הפיכת הספרה להיות הקיצונית מימין: נחלק בחזקת 10 ע"פ מיקום הספרה. לדו' במספרת תלת ספרתי, כשנרצה לגרום לספרת העשרות להיות הימנית, נחלק את המספר ב 10 בחזקת 1=10.

לדו': 4**2**9/10=4**2** ואז נוכל ללכת ע"פ הגישה של קבלת ספרה מימין- 10%. כך שהתרגיל הסופי במקרה זה יהיה כך: 4**2**9/10%10=**2**

דו' נוספת, נרצה להוציא את ספרת המאות ממספר בעל 4 ספרות: 8**9**52/100=8**9**, ואז נלך ע"פ הגישה של קבלת ספרה מימין-10%, כך: 8**9**52/100%10=**9**

הפיכת הספרה להיות הקיצונית משמאל: הפעם נשתמש בפעולת השארית % (מודולו). כיצד? נשתמש ב % ואחריו חזקת 10 התואמת למיקום הספרה משמאל לספרה שנרצה לשלוף.
לדו' במספר תלת ספרתי, נרצה לשלוף את הספרה האמצעית-ספרת העשרות, הספרה שמשמאל לה היא ספרת המאות, ולבן נשתמש ב 100%.

לדו': 4**2**9%100=**2**9, ואז נוכל ללכת ע"פ הגישה של קבלת ספרה משמאל, 10/ (כי כרגע זו ספרת העשרות) התרגיל הסופי יראה כך: 4**2**9%100/10=**2**

דו' נוספת, דו' נוספת, נרצה להוציא את ספרת המאות ממספר בעל 4 ספרות: 52%1000=**9**52, ואז נלך ע"פ הגישה של קבלת ספרה משמאל-100/, כך: 8**9**52%1000/100=**9**

סיכום

-חלוקה שלמה ב-C# מספקת רק את חלק המספר השלם, בעוד ש-Modulo (`%`) נותן את השאר.

- כדי לבצע חלוקה מדויקת בין מספרים שלמים, המירי לפחות אחד מהמשתנים בתרגיל ל double

- ניתן להשתמש בפעולות אלו כדי לחלץ ספרות ממספרים, לבדוק מאפיינים (כמו זוגי/אי זוגי), והן בסיסיות באלגוריתמים שונים.

- על ידי הבנה ויישום של מושגים אלו, ניתן לבצע פעולות אריתמטיות מדויקות ושימושיות ב-C#.