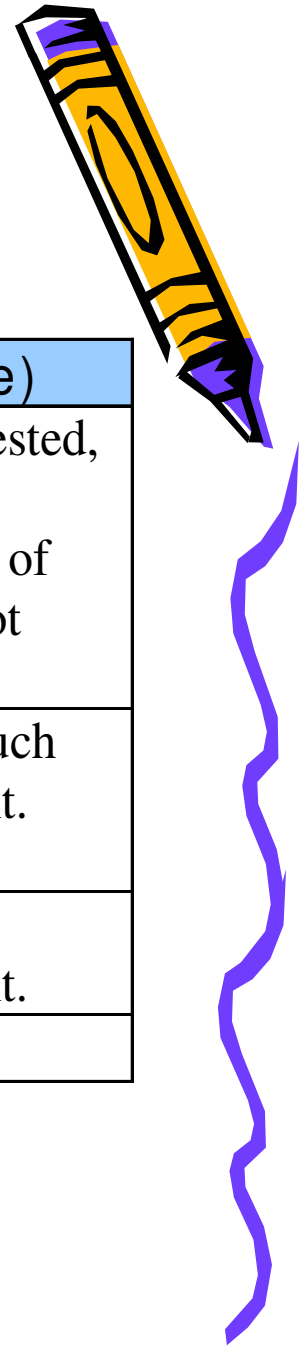
# אריתמטיקה: סדר הפעולות



| Operator(s) | Operation | Order of evaluation (precedence) |
|---|---|---|
| ( ) | Parentheses | Evaluated first. If the parentheses are nested, the expression in the innermost pair is evaluated first. If there are several pairs of parentheses "on the same level" (i.e., not nested), they are evaluated left to right. |
| *, / or % | Multiplication<br>Division<br>Modulus | Evaluated second. If there are several such operators, they are evaluated left to right. |
| + or – | Addition<br>Subtraction | Evaluated last. If there are several such operators, they are evaluated left to right. |

Precedence of arithmetic operators.

# דוגמא מפורטת

*Step 1.*

```
y = 2 * 5 * 5 + 3 * 5 + 7;
```

```
2 * 5 is 10
```

*(Leftmost multiplication)*

*Step 2.*

```
y = 10 * 5 + 3 * 5 + 7;
```

```
10 * 5 is 50
```

*(Leftmost multiplication)*

*Step 3.*

```
y = 50 + 3 * 5 + 7;
```

```
3 * 5 is 15
```

*(Multiplication before addition)*

*Step 4.*

```
y = 50 + 15 + 7;
```

```
50 + 15 is 65
```

*(Leftmost addition)*

*Step 5.*

```
y = 65 + 7;
```

```
65 + 7 is 72
```

*(Last addition)*

*Step 6.*

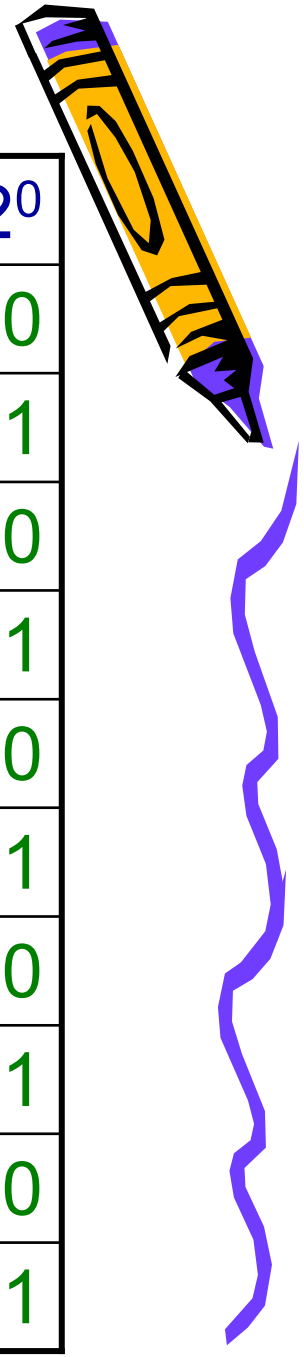
```
y = 72;
```

*(Last operation—place 72 into y)*

```
y = 2 * ((5 * 5) + 3) * 5 + 7;

    2 * (25 + 3) * 5 + 7 = 2 * 28 * 5 + 7 = 287
```

# מספרים בינאריים



| עשרוני | $2^3$ | $2^2$ | $2^1$ | $2^0$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 2 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 | 1 |
| 4 | 0 | 1 | 0 | 0 |
| 5 | 0 | 1 | 0 | 1 |
| 6 | 0 | 1 | 1 | 0 |
| 7 | 0 | 1 | 1 | 1 |
| 8 | 1 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 |

3 ספרות: $2^3$ אפשרויות

# מספרים שלמים

int: 2 bytes = 16 bits

$\pm, -(2^{15}-1) - (2^{15}-1)$

בזבוז: $\pm 0$

$2^{15}-1=32767$

2's complement:

השלמה בינארית:

דוגמא עם 3 ספרות:

$-y : 2^3-y$

+101 = 5

(-)011 : -5

סכום:

1000

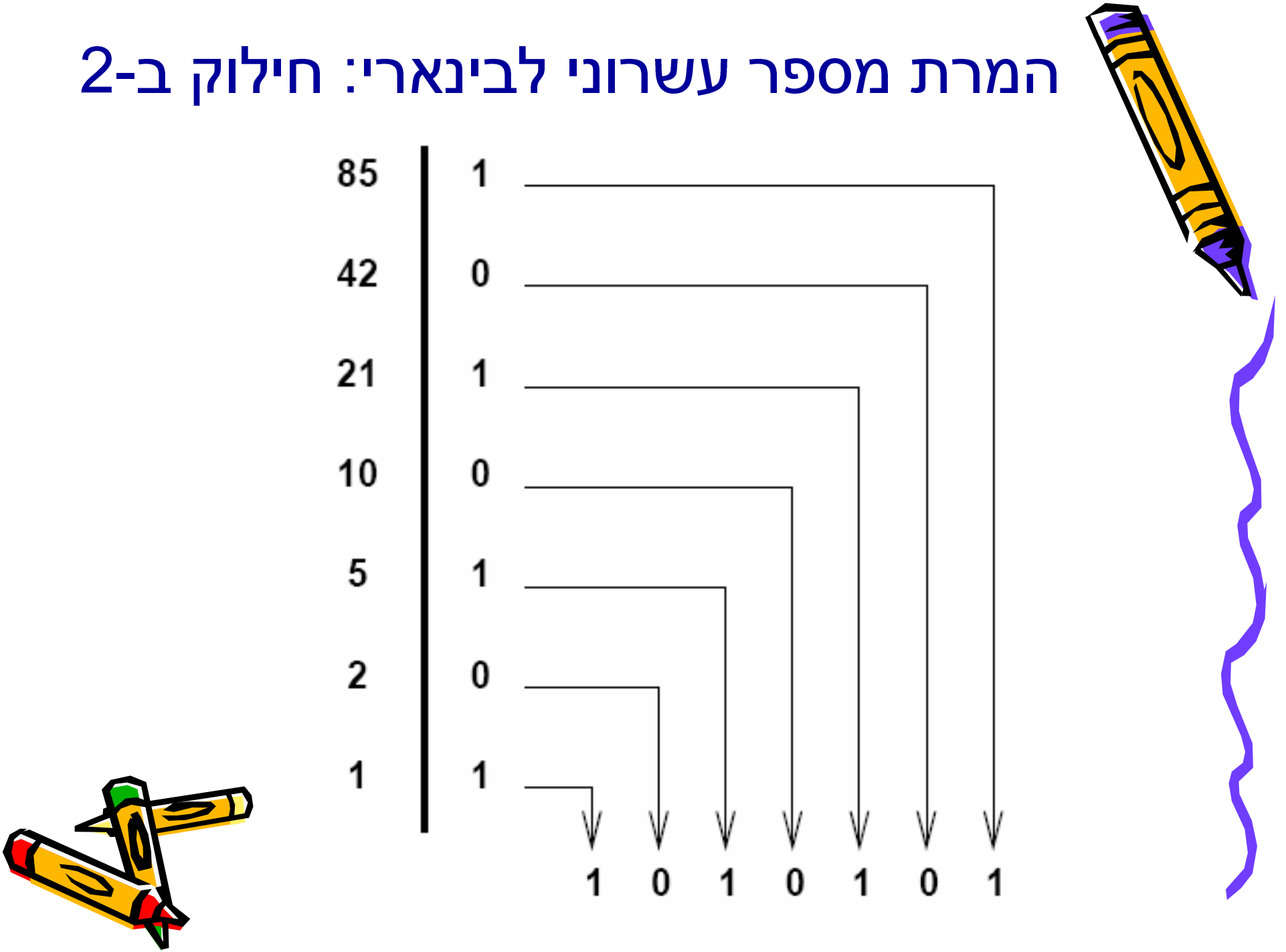
↓

0

(-)000 : -8

$\pm, -32768 - 32767$

# המרת מספר עשרוני לבינארי: חילוק ב-2



# Rounding Errors שגיאות עיגול

| | | |
|---|---|---|
| 1.2015 | 1.202 | 1.20 |
| x 3.3568 | x 3.357 | x 3.36 |
| 4.03319520 | 4.035114 | 4.0320 |

דיוק סופי:

| | | |
|---|---|---|
| 4.0332 | 4.0351 | 4.0320 |
| 4.033 | 4.035 | 4.032 |
| 4.03 | 4.04 | 4.03 |

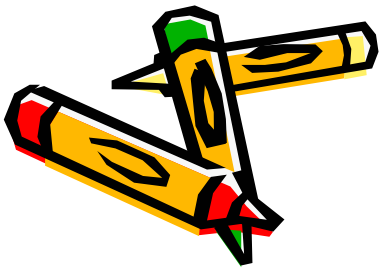
עיגול התוצאה:

דוגמא עם 3 ספרות עשרוניות:

$$600 + 600 = 1200$$

$$50 * 50 = 2500$$

overflow



$$700 + (400 - 300) = 800$$

$$(700 + 400) - 300 = 1100 - 300 = \ ?$$

# מבנה תוכנית ב-C

מעבד ראשוני:
משתנים גלובליים:

```
pre-processor directives
global declarations
```

תכנית ראשית:

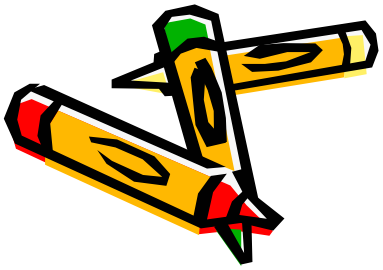
```
main( )
{
local variables to function main ;
statements associated with function main ;
}
```

פונקציה ראשונה:

```
f1( )
{
local variables to function 1 ;
statements associated with function 1 ;
 }
```

פונקציה שנייה:

```
f2( )
{
local variables to function f2 ;      /* This is a comment. */
statements associated with function 2 ;
}
 . . .
```



# מעבד ראשוני:

```
# include <stdio.h>
# include "stdio.h"
```

← מחפש בספריות

```
int i;
i=10;
i=i+10;
```

```
float sum;
sum=1.05;
```

```
char c;
c='A';
```

```
i=i+10;
i+=10;
```

```
y=y/3.;
y/=3.;
```

```
i=i+1;
++i;
i++;
```

```
i=i-1;
--i;
i--;
```