

## ▶ בפרק זה נלמד:

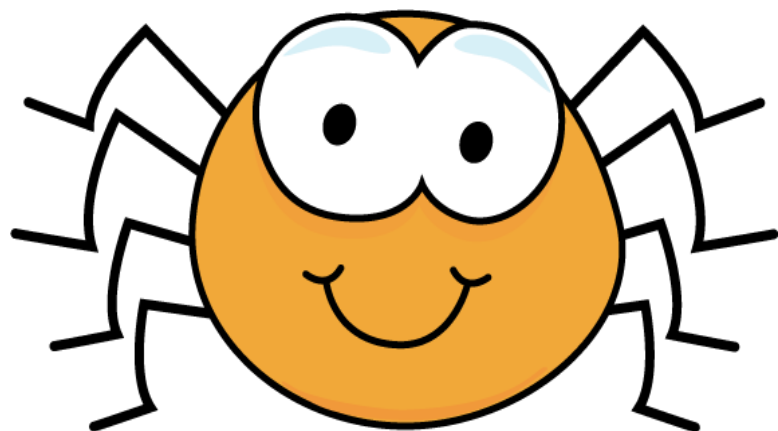
- שיטות ספירה, בדגש על בסיס 2 ו-16
- המרות בין בסיסים
- פעולות חיבור, חיסור, כפל וחילוק בבסיסים שונים
- שיטת המשלים ל-2 לייצוג מספרים Signed ו-Unsigned
- גדלים בזיכרון המחשב: ביט, Nibble, בית...
- קוד ASCII לייצוג תווים ע"י 8 ביט

- ▶ לבני אדם יש עשר אצבעות
- ▶ שיטת הספירה הדצימלית – בסיס 10 - מבוססת על עשר ספרות
  - 0,1,2,3,4,5,6,7,8,9
  - בשביל לייצג מספרים הגדולים מ-9 נדרשות שתי ספרות
- ▶ אפשר לספור גם בבסיסים אחרים!
- ▶ למה זה חשוב? בגלל האופן שבו מידע מיוצג במחשב

# ספירה בבסיסים שונים

בסיס 3 0,1,2	בסיס 8 0,1,2,3,4,5,6,7	בסיס 10 0,1,2,3,4,5,6,7, 8,9
0	0	0
1	1	1
2	2	2
10	3	3
11	4	4
12	5	5
20	6	6
21	7	7
22	10	8
100	11	9
101	12	10
102	13	11

# כמה רגליים יש לעכביש?



- ▶ בבסיס עשר - 8 רגליים
- ▶ בבסיס שמונה- 10 רגליים
- ▶ בבסיס שלוש - 22 רגליים

▶ כמות הרגליים לא השתנתה,  
רק הייצוג שלה.

# רישום בסיס הספירה

▶ מעכשיו נציין באיזה בסיס ספירה מדובר

$501_{10}$

▶ זאת כדי למנוע טעויות כגון

$10_{10}$  -----  $10_2$

$47_{10}$  -----  $47_8$

▶ הערך של ספרה נקבע לפי המיקום שלה

◦ 501 לעומת 105

▶ בבסיס עשר, ערך המיקום נקבע לפי חזקות של עשר

▶ כמה דוגמאות בבסיס עשר:

$$47_{10} = 7 \cdot 10^0 + 4 \cdot 10^1$$

$$375_{10} = 5 \cdot 10^0 + 7 \cdot 10^1 + 3 \cdot 10^2$$

$$1994_{10} = 4 \cdot 10^0 + 9 \cdot 10^1 + 9 \cdot 10^2 + 1 \cdot 10^3$$

# השיטה הבינארית - בסיס 2

- ▶ קיימות שתי ספרות בלבד - 0, 1
- המספר 2 צריך להיות מיוצג ע"י יותר מספרה אחת
- ▶ ערך המיקום של ספרה נקבע לפי חזקות של 2

$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
128	64	32	16	8	4	2	1

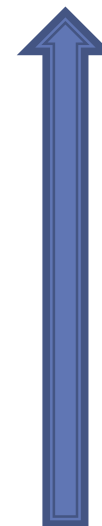
▶ לדוגמה המספר  $19_{10} = 16 + 2 + 1 = 10011_2$

$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
128	64	32	16	8	4	2	1
-	-	-	1	0	0	1	1

# המרה מעשרוני לבינארי

▶ נבצע את ההמרה ההפוכה, מ-  $19_{10}$  לבינארי

פעולה	שארית
$19:2 = 9$	1
$9:2 = 4$	1
$4:2 = 2$	0
$2:2 = 1$	0
$1:2 = 0$	1



▶ קיבלנו  $10011_2$  😊



# השיטה ההקסדצימלית - בסיס 16

- ▶ קיימות 16 ספרות
- ▶ לוקחים שש אותיות ונותנים להן ערך מספרי:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0

- ▶ מספר יכול להיות צירוף של אותיות וספרות:

- $F16_{16}$
- $C1A_{16}$
- $CODE_{16}$
- $COFFEE_{16}$

# שיטות רישום מספרים הקס'

---

- כתיבת הבסיס למטה:  $CODE_{16}$
- ▶ תוספת  $0$  לפני המספר:  $0xCODE$
- ▶ סיומת  $h$  בסוף המספר
- אם המספר מתחיל באות, מוסיפים  $0$  בתחילת המספר
- $0CODEh$

# המרה בין הקסדצימלי לדצימלי

▶ מהקסדצימלי לדצימלי:

▶  $4F_{16} = F * 16^0 + 4 * 16^1 = 15 + 64 = 79_{10}$

▶ מדצימלי להקסדצימלי:

שארית	פעולה
7	$199 : 16 = 12$
C (12)	$12 : 16 = 0$



▶ ולכן  $199_{10} = C7_{16}$

# המרה בין בינארי והקסדצימלי

הקסדצימלי	בינארי
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

▶ להמרה בין בינארי  
והקסדצימלי יש תכונה  
מיוחדת: כל ספרה הקס'  
היא ארבע ספרות בינאריות

# המרה בין בינארי והקס'-דוגמאות

$$9_{16} = 1001_2 \blacktriangleright$$

$$B_{16} = 1011_2 \blacktriangleright$$

$$9B_{16} = 10011011_2 \blacktriangleright$$

- ▶ מעבר בין בינארי להקס' הוא פשוט מאד
- ▶ כתיב בהקס' פשוט יותר לקריאה ולזכירה מאשר בינארי
- ▶ לכן הקס' שימושי בעבודה עם מחשבים

# תרגילים - מעבר בין בסיסים

בסיס 10	בסיס 2	בסיס 16
35		
	1011	
		12
	1100011	
59		
		63
		5C
	11110	
21		
		31

$$\begin{array}{r}
 1 \\
 + 133 \\
 \underline{\phantom{0}70} \\
 203
 \end{array}$$

▶ בחיבור בבסיס 10, מעבירים נשא (Carry) כשהתוצאה גדולה מ-9

$$\begin{array}{r}
 1 \\
 + 1010 \\
 \underline{\phantom{0}11} \\
 1101
 \end{array}$$

▶ בחיבור בבסיס 2, מעבירים נשא כשהתוצאה גדולה מ-1

$$\begin{array}{r}
 1 \\
 + ABCD \\
 \underline{\phantom{0}0123} \\
 ACFO
 \end{array}$$

▶ בחיבור בבסיס 16, מעבירים נשא כשהתוצאה גדולה מ-15 (0Fh)

# תרגילים - חיבור

A	B	A+B
25	F	
B	29	
26	2C	
34	34	
1F	2A	
1E	12	
22	2F	

A	B	A+B
100111	10100	
11001	11000	
110000	100101	
111001	11001	
110110	101011	
111001	10011	
111001	101010	



# ייצוג מספרים במחשב

תרגום לעשרוני	המספר הכי גדול שאפשר לייצג	N
1	1	1
3	11	2
7	111	3
15	1111	4
31	11111	5
63	111111	6
127	1111111	7
255	11111111	8

▶ כל ספרה בינארית נקראת

**Bit**: Binary Digit

▶ בזיכרון המחשב יש תאים

שבהם כמות קבועה של

ביטים

▶ תא זיכרון בגודל N ביטים

יכול לייצג מספר עד  $2^N - 1$

# ייצוג מספרים במחשב - המשך

- ▶ מה יקרה אם ננסה להכניס ל-N ביטים מספר יותר גדול ממה שניתן לייצג ע"י N ביטים?
- ▶ דוגמה:  $255 + 1$

$$\begin{array}{r}
 + 11111111 \\
 \underline{00000001} \\
 (1)00000000
 \end{array}$$

- את התוצאה אי אפשר לשמור ע"י 8 ביטים
- הזיכרון יכיל 00000000
- במקום אחר בזיכרון יודלק ביט שאומר שהיה נשא בפעולה האחרונה

## שיטת המשלים ל-2

- ▶ הנגדי של מספר הוא היפוך הביטים שלו, פלוס 1
- ▶ לדוגמה, נמצא את הנגדי של שש (00000110):

$$\begin{array}{r}
 11111001 \\
 + \phantom{11111}1 \\
 \hline
 11111010
 \end{array}$$

- ▶ שש ועוד מינוס שש:

$$\begin{array}{r}
 + 00000110 \\
 \phantom{+} 11111010 \\
 \hline
 (1)00000000
 \end{array}$$

# שיטת המשלים ל-2 - המשך

מספר עשרוני	ייצוג בשיטת המשלים ל-2	מספר עשרוני	ייצוג בשיטת המשלים ל-2
7	0000 0111	-1	1111 1111
6	0000 0110	-2	1111 1110
5	0000 0101	-3	1111 1101
4	0000 0100	-4	1111 1100
3	0000 0011	-5	1111 1011
2	0000 0010	-6	1111 1010
1	0000 0001	-7	1111 1001
0	0000 0000	-8	1111 1000

▶ בעזרת N ביטים, ניתן לייצג את התחום שבין  $-2^{N-1}$  ל-1

◦ 8 ביטים:  $-128 \rightarrow +127$

◦ 16 ביטים:  $-32,768 \rightarrow +32,767$

◦ וכו'

# שיטת המשלים ל-2 - המשך

---

▶ חסרונות:

◦ הנגדי של מספר אינו ברור כמו בשיטות האחרות

▶ יתרונות:

◦ פעולות החשבון נותנות תוצאות הגיוניות

◦ ייצוג יחיד לאפס

▶ המספרים במחשב מיוצגים בשיטת המשלים ל-2

# המרת מספר signed מבינארי לעשרוני

---

- ▶ אם המספר חיובי (ביט שמאלי 0):
  - המרה "רגילה"
  - לכל ביט יש מיקום
  - הערך נקבע לפי 2 בחזקת המיקום
- ▶ אם המספר שלילי (ביט שמאלי 1):
  - מוצאים את הנגדי של המספר
  - מוסיפים מינוס

# המרת מספר signed לבסיס עשר - דוגמה

▶ מה מייצג המספר 1011111?

- הביט השמאלי הוא 1, לכן מדובר במספר שלילי
- נמצא את הנגדי שלו ונוסיף לו סימן מינוס

מספר מקורי: 1011111

משלים ל-1: 01000000

משלים ל-2: 01000001

$$2^0 + 2^6 = 65$$

לכן המספר מייצג -65

## ייצוג signed לעומת ייצוג unsigned

- ▶ ראינו שהמספר 1011111 שווה -65
- ▶ למה שווה המספר 1011111 בייצוג unsigned?
- $128 + 32 + 16 + 8 + 4 + 2 + 1 = 191$
- ▶ אז למה שווה 1011111?
- תלוי בפרשנות שלנו! אפשר לפרש את הספרות הבינאריות כך או כך
- פרשנות כ-signed: -65
- פרשנות כ-unsigned: 191



# ייצוג מידע במחשב

---

- ▶ סיבית - Bit
- ▶ רביעיית ביטים - Nibble
- ▶ בית (8 ביטים) - Byte
- ▶ מילה (16 ביטים) - Word
- ▶ מילה כפולה (32 ביטים) - Double Word

▶ ביט שומר אחד משני ערכים:

◦ 0

◦ 1

▶ ביט יכול לייצג כל שני ערכים שונים זה מזה:

◦ 0 מייצג "שקר", 1 מייצג "אמת"

◦ 0 "ירוק", 1 "אדום"

◦ 0 "517", 1 "348"

▶ המשמעות תלויה בפרשנות שלנו

# רביעיית ביטים Nibble

הקסדצימלי	בינארי
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

- ▶ יש 16 Nibbleים שונים
- ▶ לכל Nibble מתאימה ספרה הקס'
- ▶ לכן Nibble שימושי לקריאה של רצפי ביטים

1101 1110 1010 1101 1100 0000 1101 1110  
 D E A D C 0 D E

▶ 8 ביטים

▶ היחידה הקטנה ביותר שיש לה כתובת בזיכרון

▶ סידור הביטים:



▶ ביט מספר 0 הוא ה-Low Order Bit

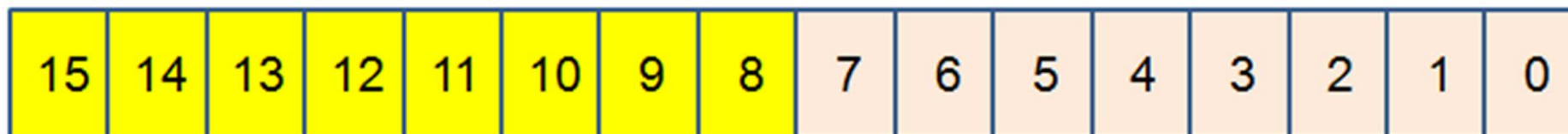
▶ ביט מספר 7 הוא ה-High Order Bit

# מילה Word

▶ 16 ביטים

▶ 2 בתים

▶ ניתן לייצג 2 בחזקת 16 = 65,536 ערכים שונים



High Order Byte

Low Order Byte

# מילה כפולה Double Word

---

▶ 32 ביטים

▶ 4 בתים

▶ שתי מילים

▶ 2 בחזקת 32 = 4,294,967,296 ערכים שונים

## American Standard Code for Information Interchange

קוד נפוץ לייצוג תווים

כל 8 ביט מייצגים תו-סה"כ 255 תווים

Regular ASCII Chart (character codes 0 - 127)															
000	<nul>	016	▶ <dle>	032	sp	048	0	064	@	080	P	096	`	112	p
001	☉ <soh>	017	◀ <dc1>	033	!	049	1	065	A	081	Q	097	a	113	q
002	☒ <stx>	018	↕ <dc2>	034	"	050	2	066	B	082	R	098	b	114	r
003	▼ <etx>	019	!! <dc3>	035	#	051	3	067	C	083	S	099	c	115	s
004	◆ <eot>	020	¶ <dc4>	036	\$	052	4	068	D	084	T	100	d	116	t
005	♣ <enq>	021	⋈ <nak>	037	%	053	5	069	E	085	U	101	e	117	u
006	♠ <ack>	022	≡ <syn>	038	&	054	6	070	F	086	V	102	f	118	v
007	• <bel>	023	‡ <etb>	039	'	055	7	071	G	087	W	103	g	119	w
008	▣ <bs>	024	↑ <can>	040	<	056	8	072	H	088	X	104	h	120	x
009	<tab>	025	↓ <em>	041	>	057	9	073	I	089	Y	105	i	121	y
010	<lf>	026	<eof>	042	*	058	:	074	J	090	Z	106	j	122	z
011	♂ <vt>	027	← <esc>	043	+	059	;	075	K	091	[	107	k	123	{
012	♀ <np>	028	└ <fs>	044	,	060	<	076	L	092	\	108	l	124	
013	<cr>	029	⊕ <gs>	045	-	061	=	077	M	093	]	109	m	125	}
014	☾ <so>	030	▲ <rs>	046	.	062	>	078	N	094	^	110	n	126	~
015	* <si>	031	▼ <us>	047	/	063	?	079	O	095	_	111	o	127	Δ

מהו הייצוג של "HELLO WORLD!"?

H E L L O      W O R L D !  
48 45 4C 4C 4F 20 57 4F 52 4C 44 21

```
ds:0000 48 45 4C 4C 4F 20 57 4F HELLO W
ds:0008 52 4C 44 21 00 00 00 00 RLD!
ds:0010 00 00 00 00 00 00 00 00
ds:0018 00 00 00 00 00 00 00 00
ds:0020 00 00 00 00 00 00 00 00
```



# תרגיל- קוד ASCII

---

▶ כיתבו את שמכם הפרטי + משפחה באנגלית,  
בקוד ASCII.

- יש לרשום גם את הקוד של תו הרווח
- אות ראשונה בשם פרטי ובשם משפחה- אות גדולה