To metoder for konvertering:

Fra desimal til andre tallsystem:

verdi	rest

1) Dele på tallsystemets base.
2) Skrive svaret i "verdi" kolonna
3) Skrive resten i "restkolonna.
   - Hvis divisjonen går opp, så er resten 0.
   - Hvis divisjonen ikke går opp får vi et desimaltall.
     → Skriv tallet før komma i "verdi" kolonna.
     → Ta forrige verdi du skrev i "verdikolonna" minus basen ganget med tallet du nettopp skrev i "verdikolonna". I
     → Dette tallet er resten. Skriv det så i "rest kolonna".
4) Gjør punkt 1) helt til siste tall i "verdikolonna" er 0.
5) Svaret er i "restkolonna". Les tallene nedenfra og opp så har du svaret ditt.

Fra binær til desimal:

binært tall: 0 1 0 1 1
base 2^4 2^3 2^2 2^1 2^0

0 + 8 + 0 + 2 + 1 = 11

1) Skriv opp det binære tallet.
2) Skriv opp basen (2) under hvert "binært siffer."
3) Start lengst til høyre på basen og sett potenser mot venstre. Start på 0.
4) Gang sammen tallene som står oppå hverandre
5) Pluss sammen alt.
6) Dette er svaret ditt i 10-tallsystemet.

Skriv svaret på denne forma: $\text{tall}_{\text{base}} = \text{svar}_{\text{base}}$

PS: Hvis du skal konvertere fra binære tall til andre tallsystem, gå gjennom 10-tallsystemet. Altså - gå fra binær til 10-tallsystemet og så til det tallsystemet du vil konvertere til.

Husk: dette er 8 bits eller 1 byte:

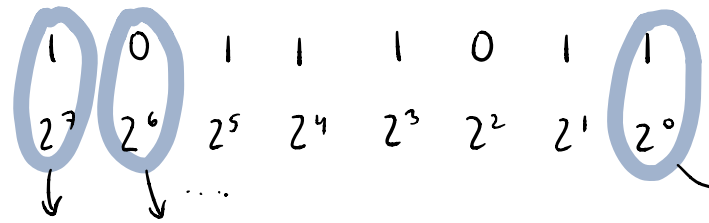
0101 1101

8 "sifre"

→ Hvis du blir bedt om å oppgi et svar med 8 bits / 1 byte, legg på så mange 0-ere som trengs for å få svaret ditt til å ha 8 "sifer"!

→ Dvs. at 0001 1101 og 11101 er det samme tallet!!!

Eksempel - fra binær til desimal



$2^7 \cdot 1 + 2^6 \cdot 0 + 2^5 \cdot 1 + 2^4 \cdot 1 + 2^3 \cdot 1 + 2^2 \cdot 0 + 2^1 \cdot 1 + 2^0 \cdot 1$

$= 128 + 32 + 16 + 8 + 2 + 1 = 187$

svaret: $10111011_2 = 187_{10}$

Eksempel - fra desimal til binær

Konverter 37_{10} fra desimal til binær og representer svaret med 8 bits.

verdi	rest
37	1
18	0
9	1
4	0
2	0
1	1
0	

svaret: $37_{10} = 100101_2 = 00100101_2$

Heksatall: når de binære tallene blir for store, så trenger vi en mer komprimert måte å skrive dem på.

Eksempel - fra binær til heksa:

Konverter 1101000111101010_2 til heksadesimal:

2 metoder:

Bruk tabellen til å konvertere. Grupper 4 og 4 siffer sammen og bruk tabellen til å oversette hver gruppe av 4 siffer til et heksatall:

- begynn lengst til høyre
- legg til 0 mot venstre om det ikke er nok i den siste gruppa.

0001	1010	0011	1110	1010
1	A	3	E	A

svaret: 1101000111101010_2

$= 1A3EA_{16} =$ 0x1A3EA

↓ betyr det er heksatall

Vanlig metode (tungvint): konverter til desimal, så til heksa

1 1 0 1 0 0 0 1 1 1 1 0 1 0 1 0

$2^{17}\ 2^{16}\ 2^{15}\ 2^{14}\ 2^{13}\ 2^{12}\ 2^{11}\ 2^{10}\ 2^9\ 2^8\ 2^7\ 2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

$= 2^{17} + 2^{16} + 2^{14} + 2^{12} + 2^9 + 2^8 + 2^7 + 2^6 + 2^4 + 2^1$

$= 107498$

vi deler på 16

verdi	rest
107498	A
6718	E
419	3
26	A
1	1
0	

svaret: $107498_{10} = 1A3EA_{16}$

så: $1101000111101010_2 = 107498_{10} = 0x1A3EA$

To av deres beste venna:

- kalkulatora som konvertera mellom tallsystem på nettet
- denne tabellen:

DECIMAL	HEX	BINARY
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Negative tall og 2-er komplement:

Hvis det står spesifikt i oppgava at vi jobba me 2-er komplement, så se på det første "sifferet" i tallet:

- Hvis 1 → tallet er negativt
- Hvis 0 → tallet er positivt

Første bit → fortegnsbit siden det forteller fortegnet
→ mest signifikantet bitet

Fra binær til desimal:

1) Det samme som når du skal regne fra binær til desimal.
2) Om fortegnsbitet er 0 → regn på vanlig måte
3) Om fortegnsbitet er 1 → negativt tall og det første bitet vil ha minus i utregninga.

Fra desimal til binær:

1) Regn ut den positive versjonen av tallet på vanlig måte slik at du får et binært tall.
2) Inverter tallene i det binære tallet
   - 0 → 1
   - 1 → 0
3) Adder dette med 1
4) Dette er svaret.

Eksempel

Vi skal konvertere 10110101_2 fra binær til desimal, der vi bruka 2'er komplement:

1	0	1	1	0	1	0	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

$$-2^7 \cdot 1 + 2^6 \cdot 0 + 2^5 \cdot 1 + 2^4 \cdot 1 + 2^3 \cdot 0 + 2^2 \cdot 1 + 2^1 \cdot 0 + 2^0 \cdot 1 =$$

$$-128 + 32 + 16 + 4 + 1 = -75$$

Svaret: $10110101_2 = -75_{10}$

PS: Hvis dere blir bedt om å skrive svaret med et visst antall bits med 2'er komplement, legg til 1-ere på starten for å få flere bits.

Eksempel

Vi skal konvertere -23_{10} fra desimal til binær med å bruke 2'er komplement og representere tallet med 8 bits.

verdi	rest
23	1
11	1
5	1
2	0
1	1
0	

$23_{10} = 00010111$

Bytter om slik at $0 \to 1$ og $1 \to 0$:

11101000

Legger til 1:

$$\begin{array}{r} 11101000 \\ +\ 00000001 \\ \hline 11101001 \end{array}$$

svaret: $-23_{10} = 11101001_2$

overflyt og antall bit:

Hvis vi skal representere et tall med 8 bit, så kan vi representere 255 ulike verdia. Hvis vi tar med negative tall kan vi skrive tall fra -128 til 127 med binærtall med 8 bits. Dette er altså alle verdia vi har plass til i 8 bits.

Overflyt er når resultatet av en beregning blir for stor til å ha plass i antall tilgjengelige bit.

- Viktig fordi vi kan få feil i beregningene våre om vi ikke tar hensyn til overflyt.

Eksempel: Vi har en prosessor med kapasitet på 8 bit. Da er det største tallet vi kan representere i binær: 1111 1111.

Vi vil gjøre en addisjon av dette tallet med 1:

$$\begin{array}{r} \overset{\leftarrow \text{mente}}{1\ 1\ 1\ 1\ 1\ 1\ 1\ 1} \\ +\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1 \\ \hline = \ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \end{array}$$

"overflyt-sifferet"

$0 + 0 = 0$
$1 + 0 = 1$
$0 + 1 = 1$
$1 + 1 = 0$ og 1 i mente

Når dette skjer, så vil prosessoren tro at resultatet av addisjonen er 0 og den vil ignorere overflyt-sifferet siden den ikke kan håndtere flere bits. Dermed blir resultatet av addisjonen feil.

ASCII TABLE

Decimal	Hexadecimal	Binary	Octal	Char
0	0	0	0	*[NULL]*
1	1	1	1	*[START OF HEADING]*
2	2	10	2	*[START OF TEXT]*
3	3	11	3	*[END OF TEXT]*
4	4	100	4	*[END OF TRANSMISSION]*
5	5	101	5	*[ENQUIRY]*
6	6	110	6	*[ACKNOWLEDGE]*
7	7	111	7	*[BELL]*
8	8	1000	10	*[BACKSPACE]*
9	9	1001	11	*[HORIZONTAL TAB]*
10	A	1010	12	*[LINE FEED]*
11	B	1011	13	*[VERTICAL TAB]*
12	C	1100	14	*[FORM FEED]*
13	D	1101	15	*[CARRIAGE RETURN]*
14	E	1110	16	*[SHIFT OUT]*
15	F	1111	17	*[SHIFT IN]*
16	10	10000	20	*[DATA LINK ESCAPE]*
17	11	10001	21	*[DEVICE CONTROL 1]*
18	12	10010	22	*[DEVICE CONTROL 2]*
19	13	10011	23	*[DEVICE CONTROL 3]*
20	14	10100	24	*[DEVICE CONTROL 4]*
21	15	10101	25	*[NEGATIVE ACKNOWLEDGE]*
22	16	10110	26	*[SYNCHRONOUS IDLE]*
23	17	10111	27	*[ENG OF TRANS. BLOCK]*
24	18	11000	30	*[CANCEL]*
25	19	11001	31	*[END OF MEDIUM]*
26	1A	11010	32	*[SUBSTITUTE]*
27	1B	11011	33	*[ESCAPE]*
28	1C	11100	34	*[FILE SEPARATOR]*
29	1D	11101	35	*[GROUP SEPARATOR]*
30	1E	11110	36	*[RECORD SEPARATOR]*
31	1F	11111	37	*[UNIT SEPARATOR]*
32	20	100000	40	*[SPACE]*
33	21	100001	41	**!**
34	22	100010	42	**"**
35	23	100011	43	**#**
36	24	100100	44	**$**
37	25	100101	45	**%**
38	26	100110	46	**&**
39	27	100111	47	**'**
40	28	101000	50	**(**
41	29	101001	51	**)**
42	2A	101010	52	*****
43	2B	101011	53	**+**
44	2C	101100	54	**,**
45	2D	101101	55	**-**
46	2E	101110	56	**.**
47	2F	101111	57	**/**
48	30	110000	60	**0**
49	31	110001	61	**1**
50	32	110010	62	**2**
51	33	110011	63	**3**
52	34	110100	64	**4**
53	35	110101	65	**5**
54	36	110110	66	**6**
55	37	110111	67	**7**
56	38	111000	70	**8**
57	39	111001	71	**9**
58	3A	111010	72	**:**
59	3B	111011	73	**;**
60	3C	111100	74	**<**
61	3D	111101	75	**=**
62	3E	111110	76	**>**
63	3F	111111	77	**?**
64	40	1000000	100	**@**
65	41	1000001	101	**A**
66	42	1000010	102	**B**
67	43	1000011	103	**C**
68	44	1000100	104	**D**
69	45	1000101	105	**E**
70	46	1000110	106	**F**
71	47	1000111	107	**G**
72	48	1001000	110	**H**
73	49	1001001	111	**I**
74	4A	1001010	112	**J**
75	4B	1001011	113	**K**
76	4C	1001100	114	**L**
77	4D	1001101	115	**M**
78	4E	1001110	116	**N**
79	4F	1001111	117	**O**
80	50	1010000	120	**P**
81	51	1010001	121	**Q**
82	52	1010010	122	**R**
83	53	1010011	123	**S**
84	54	1010100	124	**T**
85	55	1010101	125	**U**
86	56	1010110	126	**V**
87	57	1010111	127	**W**
88	58	1011000	130	**X**
89	59	1011001	131	**Y**
90	5A	1011010	132	**Z**
91	5B	1011011	133	**[**
92	5C	1011100	134	****
93	5D	1011101	135	**]**
94	5E	1011110	136	**^**
95	5F	1011111	137	**_**
96	60	1100000	140	**`**
97	61	1100001	141	**a**
98	62	1100010	142	**b**
99	63	1100011	143	**c**
100	64	1100100	144	**d**
101	65	1100101	145	**e**
102	66	1100110	146	**f**
103	67	1100111	147	**g**
104	68	1101000	150	**h**
105	69	1101001	151	**i**
106	6A	1101010	152	**j**
107	6B	1101011	153	**k**
108	6C	1101100	154	**l**
109	6D	1101101	155	**m**
110	6E	1101110	156	**n**
111	6F	1101111	157	**o**
112	70	1110000	160	**p**
113	71	1110001	161	**q**
114	72	1110010	162	**r**
115	73	1110011	163	**s**
116	74	1110100	164	**t**
117	75	1110101	165	**u**
118	76	1110110	166	**v**
119	77	1110111	167	**w**
120	78	1111000	170	**x**
121	79	1111001	171	**y**
122	7A	1111010	172	**z**
123	7B	1111011	173	**{**
124	7C	1111100	174	**|**
125	7D	1111101	175	**}**
126	7E	1111110	176	**~**
127	7F	1111111	177	*[DEL]*