Övning 1

Inledning:

Vad är talsystem, och vad har vi dem till. Poängtera att ett bitmönster inte nödvändigtvis måste representera ett binärt tal. Tolkningen gör vi baserat på mönstrets sammanhang.

Varför skall vi prata om talsystem? Det är viktigt för att förstå hur en dator fungerar.

Del 1: Talbaser

Det decimala talsystemet

Visa ett decimalt tal och visa positionerna och hur man får reda på värdet i talet.
Exempeltal: 3752

3 +	**7** +	**5** +	**2**
10^3 +	10^2 +	10^1 +	10^0
$3*10^3$ +	$7*10^2$ +	$5*10^1$ +	$2*10^0$
3000 +	700 +	50 +	2

Visa ett mindre decimalt tal med bråk.
Exempeltal: 12,25

1 +	**2** +	**2** +	**5**
10^1 +	10^0 +	10^{-1} +	10^{-2}
$1*10^1$ +	$2*10^0$ +	$2*10^{-1}$ +	$5*10^{-2}$
10 +	2 +	0,2 +	0,05

Det binära talsystemet

Visa omvandling från ett mindre binärt tal till decimalt.
Exempeltal: $1011_2 = 11_{10}$

Binärt	**1**	**0**	**1 +**	**1**
Positionsvärde	2^3 +	2^2 +	2^1 +	2^0
Utan potenser	2*2*2	2*2	2	1
Decimalt	8 +	0 +	2 +	1

Visa omvandling från ett större binärt tal till decimalt.
Exempeltal: $10\ 1101\ 1110_2 = 734_{10}$

1	**0**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**0**
2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
512	256	128	64	32	16	8	4	2	1
1*512	0*256	1*128	1*64	0*32	1*16	1*8	1*4	1*2	0*1

Decimalt till binärt

Visa ett mindre binärt tal och visa positionerna samt hur man får ut det decimala värdet.
Exempeltal: $14_{10} = 1110_2$

Operation	**Resultat**	**Rest**
14/2	7	**0**
7/2	3	**1**
3/2	1	**1**
1/2	0	**1**

Vilket ger svaret 1110_2 när man läser nedifrån.

Visa ett större binärt tal och hur man får ut det decimala värdet.
Exempeltal: $379_{10} = 1\ 0111\ 1011_2$

Rest	379	379	123	123	59	27	11	3	3	1
Värde	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Decimalt	512	256	128	64	32	16	8	4	2	1
Differens	- 133	123	- 5	59	27	11	3	- 1	1	0
Binärt	**0**	**1**	**0**	**1**	**1**	**1**	**1**	**0**	**1**	**1**

Vilket ger svaret 1 0111 1011 när man läser från höger till vänster.

Rationella tal i binärt

Visa ett exempel på binära bråktal, med positionernas värde.
Exempeltal: $11{,}011_2 = 3{,}375_{10}$

1	**1**	**0**	**1**	**1**
2^1	2^0	2^{-1}	2^{-2}	2^{-3}
1*2	1*1	0*(1/2)	1*(1/4)	1*(1/8)
2	1	0	0,25	0,125

Rationellt decimaltal till binärt

Exempeltal: $13{,}375_{10} = 1101{,}011_2$

Rest	13,375	5,375	1,375	1,375		0,375	0,375	0,12 5
Värde	2^3	2^2	2^1	2^0		2^{-1}	2^{-2}	2^{-3}
Decimalt	8	4	2	1	,	0,5	0,25	0,12 5
Differens	5,375	1,375	-0,625	0,375		- 0,125	0,125	0
Binärt	**1**	**1**	**0**	**1**		**0**	**1**	**1**

Rationella tal som inte går att representera binärt

Vissa tal går helt enkelt inte att representera binärt eftersom det inte finns någon kombination av

värden efter kommat som ger det exakta tal man är ute efter, exempelvis 0,1.
Det är variabelt hur detaljerad man vill vara här.

1/16 + 1/32 + 1/64 = 0,0625 + 0,03125 + 0,015625 = 0,109375
1/16 + 1/32 +1/128 = 0,1015625
1/16 + 1/32 + 1/256 = 0,09765625
1/16 + 1/32 + 1/256 + 1/512 = 0,099609375
1/16 + 1/32 + 1/256 + 1/512 + 1/1024 = 0,1005859375
1/16 + 1/32 + 1/256 + 1/512 + 1/1024 + 1/2048 = 0,1000976563
1/16 + 1/32 + 1/256 + 1/512 + 1/1024 + 1/4096 = 0,09985351562
1/16 + 1/32 + 1/256 + 1/512 + 1/1024 + 1/4096 + 1/8192 = 0,09997558594
1/16 + 1/32 + 1/256 + 1/512 + 1/1024 + 1/4096 + 1/8192 + 1/16384 = 0,1000366211
Och så vidare eftersom det inte går att representera $0,1_{10}$ binärt.

Vissa tal, som till exempel ⅓, går att representera exakt. ⅓ binärt är $2^{-2} + 2^{-4} + 2^{-8} + 2^{-16} + \ldots + 2^{-(n \in \{\text{jämna tal}\})}$, men där faller man i regel på att man faktiskt inte har oändligt antal tecken att representera ett tal med. Värt att notera är att ⅓ inte går att representera exakt i det decimala talsystemet heller, utan det blir 1,333 … 3.

Det handlar om antalet primfaktorer i basen, och varken 10 eller 2 har särskilt många primfaktorer. Om man, till exempel, skulle ha basen 30 [2*3*5] skulle man kunna representera många fler tal exakt.

Det hexadecimala talsystemet

Varför använder vi det hexadecimala talsystemet? För att det är ett kompakt sätt att uttrycka tal som datorn kan jobba med. Att översätta binära tal till hexadecimala och vice versa är väldigt enkelt.

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

Visa ett mindre hexadecimalt tal och visa positionerna samt hur man får ut det decimala värdet.
Exempeltal: $2B_{16} = 43_{10}$

Hexadecimalt	**2**	**B**

Positionsvärde	16^1	16^0
Utan potenser	16	1
Decimalt	2*16	11*1

Visa ett större hexadecimalt tal och hur man får ut det decimala värdet.
Exempeltal: $CAFE_{16} = 51966_{10}$

C	A	F	E
16^3	16^2	16^1	16^0
4096	256	16	1
12*4096	10*256	15*16	14*1
49152	2560	240	14

Omvandling från decimalt till hex

Visa omvandling från ett mindre decimalt tal till hex.
Exempeltal: $27_{10} = 1B_{16}$

Operation	Resultat	Rest
27/16	1	11 = **B**
1/16	0	**1**

Vilket ger $1B_{16}$ om man läser nedifrån och upp.

Visa omvandling från ett större decimalt tal till hex.
Exempeltal: $4412_{10} = 113C_{16}$

Operation	Resultat	Rest
4412/16	275	12 = **C**
275/16	17	**3**
17/16	1	**1**
1/16	0	**1**

Vilket ger $113C_{16}$ om man läser nedifrån och upp.

Med tabell:

Rest	4412	316	60	12
Positionsvärde	$16^3 = 4096$	$16^2 = 256$	$16^1 = 16$	$16^0 = 1$

Antal gånger	1	1	3	12
Differens	4412- (1*4096) = 316	316-(1*259) = 60	60-(3*16) = 12	0
Hex	**1**	**1**	**3**	**C**

Från decimalt till hex via binärt

Att direkt omvandla mellan det decimala och hexadecimala talsystemet kräver ordentligt med huvudräkning eftersom man måste dela med 16. Ett smidigare sätt att att istället först omvandla till det binära talsystemet.
Exempeltal: $4412_{10} = 113C_{16}$

Operation	**Resultat**	**Rest**
4412/2	2206	0
2206/2	1103	0
1103/2	551	1
551/2	275	1
275/2	137	1
137/2	68	1
68/2	34	0
34/2	17	0
17/2	8	1
8/2	4	0
4/2	2	0
2/2	1	0
1/2	0	1

Vilket ger **1 0001 0011 1100** när man läser uppifrån och ned. Nästa steg blir att omvandla det binära talet till hex. Det gör man genom att dela in talet i delar om fyra bitar, så att det maximala värdet för varje del är 15 (vilket är det maximala värdet för varje position i det hexadecimala talsystemet).

Binärt	0001	0001	0011	1100
Decimalt	1	1	3	12
Hex	1	1	3	C

Notera att vi inte adderar ihop talen här, utan bara skriver upp dem efter varandra för att få det hexadecimala värdet. Vi har också lagt på tre nollor för den första biten för att det ska synas att man alltid betraktar det som fyra bitar. Inledande nollor kan man naturligtvis ta bort, men bara de som inleder talet till vänster.

Rationella tal i hex

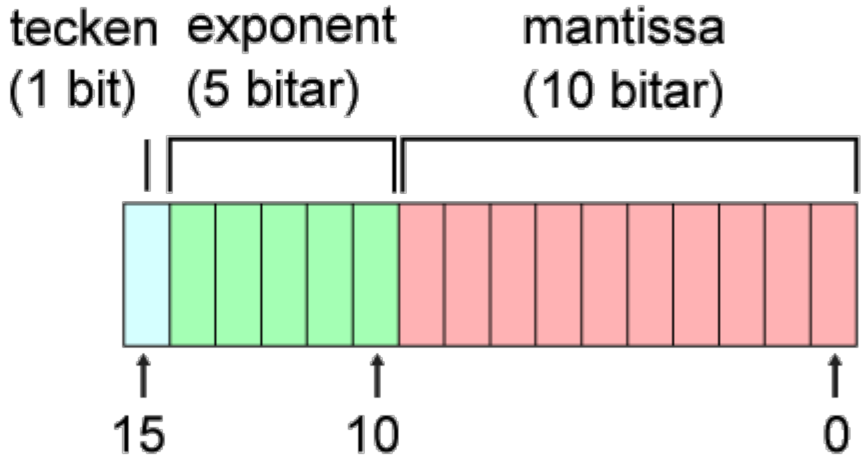
Visa ett exempel på hexadecimala bråk, med positionernas värde.
Exempeltal: $B{,}12_{16} = 11{,}0703125_{10}$

Hex	**B**	**1**	**2**
Positionsvärde	16^0	16^{-1}	16^{-2}
Utan potenser	1	1/16	1/256
Decimalt	11*1	1*(1/16)	2*(1/256)
Totalt	11	0,0625	0,0078125

Flyttal

Flyttal är ett sätt att representera rationella tal som kan anta ett stort spann av värden. Det finns flera standarder för flyttal, men i den här kursen kommer vi att använda IEEE 754.

IEEE 754 finns i flera varianter beroende på antal bitar, men vi kommer att använda 16 bitars flyttal som exempel. Ett 16 bitars flyttal är indelat i följande fält:



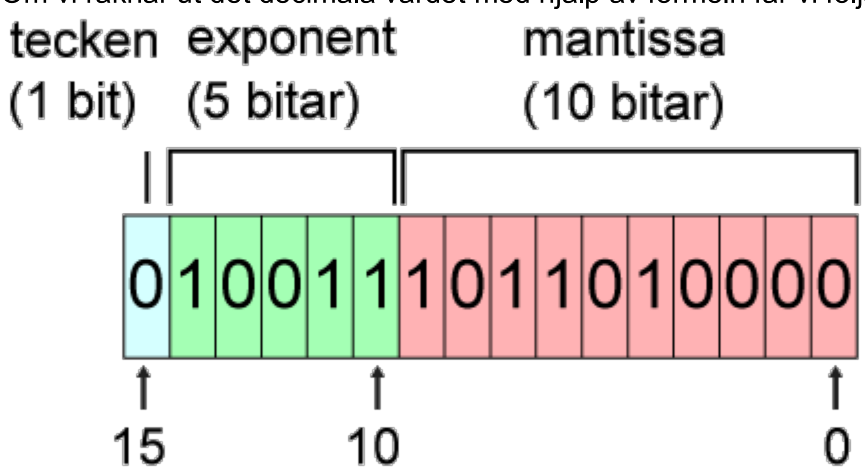
För att räkna ut värdet *v* för ett flyttal kan vi använda formeln

$$v = (-1)^{\text{teckenbit}} \times 2^{\text{exponent-15}} \times (1,\text{ mantissa})_2$$

Exponenten - 15 kommer från att vi har 5 bitar för exponenten: $2^5/2 - 1 = 32/2 - 1 = 16 - 1 = 15$. På det viset kan man representera tal som är både större och mindre än 1, eftersom om exponentbitarna ≥ 15 får man en positivt exponent, vilket ger ett flyttal > 0. Om man istället har ett värde i exponentbitarna < 15 får man en negativ exponent, vilket kommer att flytta decimalkommat så att flyttalet < 0.

Exempeltal: 0 10011 1011010000

Om vi räknar ut det decimala värdet med hjälp av formeln får vi följande steg:



Steg 1: Kontrollera om talet är positivt eller negativt:
$(-1)^{teckenbiten} = (-1)^0 = 1$. Talet är därmed positivt eftersom vi får 1*resten av formeln.

Steg 2: Beräkna värdet i exponentbitarna, vilket ger $10011_2 = 19_{10}$

Steg 3: Beräkna exponenten - 15, vilket är 19 - 15 = 4.

Steg 4: Flytta decimalkommat så många steg som exponenten anger (vilket är 4):
$1{,}101101_2 \rightarrow 11011{,}01_2 = 27{,}25_{10}$

Exempeltal: 1 01110 100000000
Steg 1: Kontrollera om talet är negativt eller positivt: $-1_1 = -1$, vilket ger ett negativt tal.
Steg 2: Beräkna värdet i exponentbitarna: $01110_2 = 14_{10}$
Steg 3: Beräkna exponenten - 15: 14 -15 = -1
Steg 4: Låt decimalkommat flyta: $-1{,}1_2 \rightarrow -0{,}11_2 = -0{,}75$

Exempeltal: $27{,}25_{10}$ och omvandla till flyttal.
Steg 1: Omvandla 27,25 till binärt: $11011{,}01_2$
Steg 2: Bestäm teckenbiten: $27{,}25 > 0 \rightarrow$ teckenbiten $= 0$.
Steg 2: Normera mantissan. Enligt formeln ska vi ha 1,mantissan, vilket ger att vi får göra om 11011,01 till 1,101101. Vi räknar och ser att vi flyttat decimalkommat fyra steg.
Steg 3: Beräkna exponentvärdet: 4 + 15 (eftersom man tar -15 i formeln) = $19_{10} = 10011_2$
Steg 4: Lägg ihop och få ett binärt flyttal: 0 10011 101101