Cheatography

Tolvuhogun Cheat Sheet

by dewey165 via cheatography.com/20059/cs/2926/

Skel leiðbeiningar

Skel Command	Hvað það gerir:
touch	create
tar	compression
grep pattern files	search for pattern in files
rm -r	remove directory
tail	output last 10 lines of file
xf - cf	extract - create
grep -r	search recursively in dir
rm	delete file
man	Manual, get help
info	documentation, replaces man erfiðara að nota
pwd	show current working directory
cd	change current directory
ls	show lists of files or information
cp	copy file
mv	move file or rename it
mkdir	make directory
rmdir	delete directory
wc	print byte,word and line counts
sort	Sort text files
cat	Send a file to the screen in one go
cat *.txt > outfile	add all .txt files together
less	Display output one screen at a time

Skel leiðbeiningar (cont)

head	Output the first part of file
uniq	checking for uniqueness, leita eftir eiginleikum t.d. nafn
chmod	change access permissions
cut	Divide a file into several parts
more	show a file one screen at a time
./	run a file

Bitwise operator

AND &	1001 AND 0101 = 0001
OR |	1001 OR 0101 = 1101
NOT ~	NOT 1001 = 0110
XOR ^	1001 XOR 1011 = 0010

Precision F10

Single precision 32 bits	1 s - exp 8bits - frac 23bits
Double precision 64 bits	1 s - exp 11bits - frac 52bits

Floating point

Case 1	Case 2	Case 3
Normalized values	**De-normalized values**	**Special values**
Exponent field is *neither* all-zero nor all-one	Exponent field is **all-zero**	sértilfelli deilt með 0 og óendaleikinn
E = e - bias	**E = 1 - bias**	Exponent = **all ones**

Floating point (cont)

e is an integer codedon is between 0 - 1 in binary(exp field)	101 = 5/2^3	Mantissa = **non-zero**
M = 1 + f	M = f	**infinity** = expo *all-ones* Mantissa *all-zero*

Data types

C data types	Assembly equilvalent	Assembly suffix	Size in bytes
char	byte	b	1
short	word	w	2
int	double word	l	4
char *	double word	l	4
float	single precision	s	4
double	double precision	l	8
long double	Extended precision	t	10/12

GCC compiler

gcc -O1 -S -m32 -0 sum.s sum.c	source file *sum.c*, output file *sum.s*, level 1 optimizations, 32 bit assembly code

Hw3 problem 1

Dæmi a : **((a ^ b) & ~b) | (~(a^b)&b)**

(x&~b) | (~x&b) = x^b // *x = a^b*

(a^b) ^ b = a ^ b ^ b =**a**

Dæmi b: **1 + (a << 3) + ~a**

~a + 1 + (a<<3)

-a + (a<<3)

-a + 8*a = **7a**

Dæmi c : **~(~a|(b^(MIN_INT+MAX_INT)))**

~(~a |(b ^**-1**)) = ~(~a|~b) = **a&b**

Dæmi d : **(a<<4) + (a<<2) + (a<<1)**

a*16 + 4*a + 2*a = **22a**

Dæmi e : **a ^ (MIN + MAX)**// *(MIN + MAX) = -1*

a ^ -1 = **~a**

Dæmi f : **~((a |(~a+1)) >> W)&1**

1XXX | 0XXX = 1XXX \\ 0000 | 0000 = 0000

1XXX >> W == 1111 \\ 0000 >> W == 0000

~(1 >> W)&1 \\ 0 & 1 = 0\\ 1 & 1 = 1\\ **(a == 0)**

Dæmi g : **((a<0)?(a+3): a)>>2**

a / 4 gerir ekki rad fyrir minus tolum

Dæmi h : **~((a >> W) << 1)**

~(sign << 1)

~1111 << 1 // negative

0001 = 1

~0000 << 1 // positive

1111 = -1

Dæmi i : **a >> 2**

gerdur til ad rugla thvi hann tekur ekki minus tolur

Problem 2

Number	Decimal	Binary
Zero	0	00000
n/a	-4	11100
n/a	11	01011
n/a	-14	10010
n/a	14	01110
n/a	-11	10101
TMax	15	01111
TMin	-16	10000
TMin + TMin	0	00000
TMin + 1	-15	01111
TMax + 1	-16	10000
-TMax	-15	10001
-TMin	-16	10000
Mínus tölur	~10000 + 1	01111 + 1

Problem 3

Description	*Hex*	*m*	*E*
-0	8000	M = 0/256	1 - bias = -62
smallest value > 1		256/256 + 1/256 = **257/256**	E = e - bias = 0
Largest Denormalized	255*2^(-70)	255/256	E = 1 - bias(63) = -62
Hex 3AA0		m = 1+fraction/2^8	E = e - bias = 58 -63 = -5

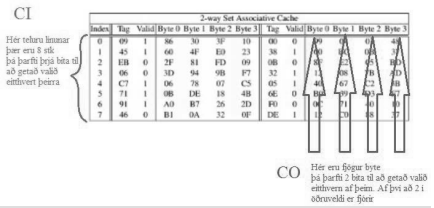
Problem 6

foo:
pushl %ebp
movl %esp,%ebp
movl 8(%ebp),%ecx \\ ecx = *a

Problem 6 (cont)

movl 16(%ebp),%edx \\edx = val
movl 12(%ebp),%eax \\ eax = n
decl %eax \\ eax = n-1
js .L3 \\ if(n-1 < 0) goto .L3
.L7:
cmpl %edx,(%ecx,%eax,4) \\ temp = a[i] - val
jne .L3 \\ if (a[i] != val) goto .L3
decl %eax \\ eax = i-1
jns .L7 \\ if(i >= 0) goto .L7
.L3:
movl %ebp,%esp
popl %ebp
ret
int foo(int *a, int n, int val) {
int i;
for (i = **_n-1__**; **_(a[i] == val) && (i >= 0)_** ; i = **_i - 1_**)
{
;
}
return i;
}

cache



HW4 problem 3

copy_element:
pushl %ebp
movl %esp,%ebp
pushl %ebx
movl 8(%ebp),%ecx // %ecx = **i**
movl 12(%ebp),%ebx // %ebx = **j**
movl %ecx,%edx // %edx = **i**
leal (%ebx,%ebx,8),%eax // %eax = **8j** + **j** = **9j**
sall $4,%edx // %edx = **16i**
sall $2,%eax // %eax = **36j**
subl %ecx,%edx // %edx = **16i** - **i** = **15i**
movl mat2(%eax,%ecx,4),%eax // %eax = mat2 + **36j** + 4**i**
sall $2,%edx // %edx = **60i**
movl %eax,mat1(%edx,%ebx,4) // mat1 + **60i**+ 4**j** = %eax
movl -4(%ebp),%ebx //
movl %ebp,%esp //
popl %ebp
ret

HW4 problem 3 (cont)

int **mat1[M][N]**
int **mat2[N][M]**
mat1[i][j] = mat2[j][i]
mat2 =36j = 36 = M
mat1 = 144 = N