Michael McCann
CSCI-C335
HW2
P.1

1. Prove, using induction that for radix-r, the largest number that can be represented with N digits is $r^N - 1$.

Proof: Base case: $N=0$, then the largest number that can be represented is $r^0 - 1 = 1 - 1 = 0$, which is vacuously true.
let $N=1$, then $r^1 - 1 = r - 1$, which is true due to there are r options and the first is 0, which means the max is $r-1$.
Assume this holds true for $N=k$, $k \in \mathbb{N}$, $k \geq 1$.
Then largest number is $\sum_{i=0}^{k-1} (r-1) r^i$
$= (r-1) r^0 + (r-1) r + (r-1) r^2 + (r-1) r^3 + \cdots (r-1) r^{k-1}$
$= r^k - 1$.
Prove this holds true for som N, $N = k+1$.
if $N=k+1$, then $\sum_{i=0}^{k} (r-1) r^i$
$= (r-1) r^0 + (r-1) r + (r-1) r^2 + (r-1) r^3 + \cdots (r-1) r^{k-1} + (r-1) r^k$
$= r^k - 1 + (r-1) r^k$, by IH
$= r^k - 1 + r^{k+1} - r^k$
$= r^{k+1} - 1$, thus this holds true $\forall N \geq 0$.
Therefore the largest radix-r number with N digits is $r^N - 1$.

Michael
McCann

2. Prove that radix-r addition, the carry bits are always 1 or 0.

Radix-r

<u>Proof by contradiction:</u> Assume there is some 2 dits when added together cary is larger than 1, then $n_1 + n_2 = xk$, where $n_1, n_2, x, k < r$, $n_1, n_2, k \geq 0$, and $x > 1$. Then this meas $n_1 + n_2 \geq 2r$. This cant be since $n_1, n_2 < r$, thus we can say $r-1 + r-1 \geq 2r$

$2r - 2 \geq 2r$

$-2 \geq 0$, which is false, thus any 2 digits of radix-r numbers when added will have a cary of no more than 1. We can use the same proof to show that any 2 digits of radix-r numbers when added with a previous cary of 1, will be $n_3 + n_4 + 1 \geq 2r$

$r-1 + r-1 + 1 \geq 2r$

$2r - 1 \geq 2r$

$-1 \geq 0$, which is false.

Thus any 2 digits of radix-r, with any possible cary over will again be at most one.

showing cary over of 1 and 0 is possible. Take base 10 $9 + 9 = 18$, which is a cary of 1, or take $5 + 3 = 8$, which is a cary of 0. Thus a cary of 1 and 0 is possible.

Therefore in radix-r addition, the cary bits are always 1 or 0.

Michael
McCann

3. Give the formal definition, derive the minimum and maximum twos complement with N bits.

<u>Definition:</u> $(b_{N-1} \cdots b_N)_2$ is for each bit, where $K = N-2$ b_K is 0 if $b_K = 1$, else 1. Then add 1.

The max will be when the last $N-1$ bits are 0 to begin with, which means we have $\underbrace{0000\cdots0}_{N-1 \text{ times}}$, then 2's complement will be

$$(\underbrace{1111\cdots1}_{N-1 \text{ times}}) + 1 = 1\underbrace{0000\cdots0}_{N-1 \text{ times}}$$

, which translates to a minimum of -2^{N-1}.

The maximum will be when the last $N-1$ bits are 1 and the N^{th} bit is 0, which means positive in 2's complement form. Thus the max is $0\underbrace{1111\cdots1}_{N-1 \text{ times}}$ or $2^{N}-1$.

Michael
McCann

4. For a number B with magnitude les than 2^{N-2}, show that if B is represented by 2's complement with N bits $b_{N-1} \cdots b_0$, then $-(b_{N-1} \cdots b_0) = (\overline{b_{N-1} \cdots b_0})_2 + 1$.

By definition 2's complement the left bit determins if something is positive or negative. Then a number that is of magnitude of less than 2^{N-1} will be $00 \underbrace{b_{N-3} b_{N-4} \cdots b_0}_{N-2 \text{ bits}}$.

Notice the complement will be $11 \underbrace{\overline{b_{N-3}}\ \overline{b_{N-4}} \cdots \overline{b_0}}_{N-2 \text{ bits}}$.

When we add 1 one we will be a magnitude of at most $2^N - 1$, except for when all of the $b_{N-3} \cdots b_0$ bits are 0 and in that case we will have $100\underbrace{00 \cdots 0}_{N-2 \text{ bits}}$, which when we keep N bits we now have $00\underbrace{00 \cdots 0}_{N-2 \text{ bits}}$, which is 0, which $-0 = 0$, thus this still works. Any other time this still holds true, by def 2's complement.

Michael
McCann

5. Prove that "sign-extension" is value preserving.

The sign extension is value preserving, due to only the N-1 bits in a number, in a N bit storage, are used, which allow for the last N^{th} bit to be the sign, which in 2's complement to be the negitive numbers. This does not hold true when adding positive numbers all the time though, for example take an 8 bit number, where the 8^{th} bit is the sign and add the max positive with 1, we would have

$$\begin{array}{r} 01111111 \\ 00000001 \\ \hline 10000000 \end{array}$$

, which is -128 not 128, which is what was expected, due to overflow.

Michael
McCann

6. Convert the decimal numbers into binary:

a) 121 Repeated Division (Mod 2)

Quotients	121	60	30	15	7	3	1
Remainders	1	0	0	1	1	1	1

$(121)_{10} = (1111001)_2$

b) 1537

Quotients	1537	768	384	192	96	48	24	12	6	3	1
Remainders	1	0	0	0	0	0	0	0	0	1	1

$(1537)_{10} = (11000000001)_2$

c) 31333

Quotients	31333	15666	7833	3916	1958	979	489	244	122	61	30	15	7	3	1
Remainders	1	0	1	0	0	1	1	0	0	1	0	1	1	1	1

$(31333)_{10} = (111101001100101)_2$

d) 97

Quotients	97	48	24	12	6	3	1
Remainders	1	0	0	0	0	1	1

$(97)_{10} = (1100001)_2$

7. a) 121-41

$$\begin{array}{r} 121 \\ -\ 041 \\ \hline 80 \end{array} \Rightarrow 121 + (\overline{41} + 1)$$

$\overline{41} + 1 = 058 + 1 = 959$

121 - 41 =

normal →

$\boxed{1\ 0\ 1}$ ← carry

$$\begin{array}{r} 121 \\ +\ 959 \\ \hline 080 \end{array}$$

, which since we are left with one carry we get $121 - 41 = 80$

Michael McCann

P. 7

7. Cont.

b) 1022-35

$$\begin{array}{r} 1022 \\ -0035 \\ \hline 987 \end{array}$$ ← normal

$\overline{0035} = 9964$

$$\begin{array}{r} 9964 \\ +\quad 1 \\ \hline 9965 \end{array}$$

$\boxed{1000}$ ← cary

$$\begin{array}{r} 1022 \\ +9965 \\ \hline 0987 \end{array}$$

since there is one cary, we get
1022-35 = 987

c) 151-90

$$\begin{array}{r} 151 \\ -90 \\ \hline 61 \end{array}$$ ← normal

$\overline{090} = 909$

$$\begin{array}{r} 909 \\ +\quad 1 \\ \hline 910 \end{array}$$

$\boxed{100}$ ← cary

$$\begin{array}{r} 151 \\ +910 \\ \hline 061 \end{array}$$

since there is one cary, we get
151-90 = 61

d) 2120-101

$$\begin{array}{r} 2120 \\ -101 \\ \hline 2019 \end{array}$$ ← normal

$\overline{0101} = 9898$

$$\begin{array}{r} 9898 \\ +\quad 1 \\ \hline 9899 \end{array}$$

$\boxed{1110}$ ← cary

$$\begin{array}{r} 2120 \\ 9899 \\ \hline 2019 \end{array}$$

since there is one cary, we get
2120-101 = 2019

8. Convert the following numbers to 8-bit 2's complement.

a) -121 $(121)_{10} = (1111001)_2$

Quotent	121	60	30	15	7	3	1
remainder	1	0	0	1	1	1	1

$\overline{(01111001)_2} = 10000110$

$$\begin{array}{r} 10000110 \\ +\quad 1 \\ \hline 10000111 \end{array}$$

$-(121)_{10} = (10000111)_2$

b) -51 $(51)_{10} = (110011)_2$

Quotent	51	25	12	6	3	1
Remainder	1	1	0	0	1	1

$\overline{(00110011)_2} = 11001100$

$$\begin{array}{r} 11001100 \\ +\quad 1 \\ \hline 11001101 \end{array}$$

$-(51)_{10} = (11001101)_2$

Michael McCann
P.8

8. Cont.

c) -104

$(104)_{10} = (1101000)_2$

$(01101000)_2 = 10010111$

$$\begin{array}{r} 10010111 \\ + \quad\quad\quad 1 \\ \hline 10011000 \end{array}$$

$(-104)_{10} = (10011000)_2$

Quotent	104	52	26	13	6	3	1
Remainder	0	0	0	1	0	1	1

d) 115

$(115)_{10} = (1110011)_2$

$(115)_{10} = (01110011)_2$

Quotent	115	57	28	14	7	3	1
Remainder	1	1	0	0	1	1	1

e) 127

$(127)_{10} = (1111111)_2$

$(127)_{10} = (01111111)_2$

Quotent	127	63	31	15	7	3	1
Remainder	1	1	1	1	1	1	1