Q1

(a)

		R
$2\overline{)47}$	= 23	1
$2\overline{)23}$	= 11	1
$2\overline{)11}$	= 5	1
$2\overline{)5}$	= 2	1
$2\overline{)2}$	= 1	0
$2\overline{)1}$	= 0	1

$+47_{10} = 00101111_2$

8-bit 2s complement

$0|0|1|0|1|1|1|1|_2 = 2F_{16}$

(b)

		R
$2\overline{)13}$	= 6	1
$2\overline{)6}$	= 3	0
$2\overline{)3}$	= 1	1
$2\overline{)1}$	= 0	1

$+13_{10} = 00001101$ (−)

11110010

$-13_{10} = 11110011_2$ (+1)

$= F3_{16}$

(c) $47_{10} = ?$

$47_{10} = 101111_2$

Normalize: $101111_2 = 1.01111_2 \times 2^5$

Mantissa: 01111000000000000000000

exponent: $5 + 127 = 132_{10} = 10000100_2$

sign bit: 0

47_{10} is 0|10000100|01111000000000000000000

$= 423C0000_{16}$

(d) $-0.375_{10} = ?$

a. integral part is 0

$0.375 \times 2 = 0.75$ 0

$0.75 \times 2 = 1.5$ 1

$0.5 \times 2 = 1$ 1

$0.375_{10} = 0.011_2$

b. normalize $0.011_2 = 1.1 \times 2^{-2}$

c. mantissa is 10000000000000000000000

exponent is $-2 + 127 = 125 = 0111101_2$

sign bit is 1

$-0.375_{10} = 1 | 0111101 | 10000000000000000000000$

$= BEC00000_{16}$

(e) "String for 250!"

ASCII dec: 83 116 114 105 110 103 32 102 111 114 32 50 53 48 33

Hex: 53 74 72 69 6E 67 20 66 6F 72 20 32 35 30 21

(f) any # higher than $2^{32}-1$

i.e. $4,294,967,297 = (2^{32}+1)$

a 32-bit computer can't store a number that can't be represented by only $2^{32}-1$ bits.

Q2

(a) a. stack b. heap c. heap d. global e. stack

(b) main() returns 0

Q3

	Unoptomized	Optomized
real	1.219s	0.354s
user	1.195s	0.372s
sys	0.009s	0.007s

User optomized speeds is 3.5 times faster!

for both the real and user times, the optomized code was over 3x faster than the unoptomized code, and the sys speed was 0.002s quicker optimized