1.

Line Number	PC	Instruction	%rdi	%rsi	%rax	%rsp	*%rsp	Description
14	4004fb	Callq	3	9	0	0x7fffffffe820	---	Call top(x,y)

6	4004dd	Add	3	9	0	0x7fffffffe818	400500	Entry of top
7	4004e0	Mov	12	9	0	0x7fffffffe818	400500	Move 0
8	4004e5	Callq	12	9	0	0x7fffffffe818	400500	Call leaf

2	4004d6	Move	12	9	0	0x7fffffff810	4004ea	Entry leaf
3	4004d9	Sub	12	9	12	0x7fffffff810	4004ea	Sub y-z
4	4004dc	Retq	12	9	3	0x7fffffff810	4004ea	Return leaf

9	4004ea	Repz retq	12	9	3	0x7fffffff818	400500	Return top

15	400500	Repz retq	12	9	3	0x7fffffffe820	--	Return main

2. 1. 48 bytes
   2. Lines 24 and 8b.
   3. Local variables d and e are stored in rbx and rbp.
   4. Variables a, b, and c are stored on the stack.
   5. Only needed the callee registers to hold 2 values while stack does the arithmetic. We could use the 6 callee registers to hold the variables; however, we would need one more register to hold some values. We just chose not to store the values.

3.
```
long unknown(unsigned long x) {
        if ( x == 0)
                return 0;
         unsigned long nx = ((1 ^ x) & 1);
        long rv = unknown(x);
         return x + rv;
}
```

   ->%Rbx stores the result of x xor 1 and x and 1.

4.

	Array Declaration	Element size	Total size	Start address	Element i
A	char r[4];	1	4	x_r	x_r + i
B	char *s[4];	8	32	x_s	x_s + 8i
C	short t[5];	2	10	x_t	x_t + 2i
D	short *u[5];	8	40	x_u	x_u +8i
E	short **v[3];	8	24	x_v	x_v + 8i
F	int w[4];	4	16	x_w	x_w + 4i
G	long *x[5];	8	40	x_x	x_x + 8i
H	double *y[6];	8	48	x_y	x_y + 8i

5.

	Expression	Type	Value	Assembly Code
A	S[2]	Short	M[x_s + 4]	movw 4(%rdx), %ax
B	S + 2	Short*	x_s + 4	leaq 4 (%rdx), %rax
C	&S[i]	Short*	x_s + 2i	leaq (%rdx, %rcx, 2), %rax
D	S[2*i+1]	Short	M[x_s +4i +2]	movw 2 (%rdx, %rcx, 4), %ax
E	S+i-2	Short*	x_s + 2i -4	leaq -4 (%rdx, %rcx, 2), %rax
F	*(S+i+2)	Short	M[X_s + 2i -4]	movw -4 (%rdx, %rcx, 2), %ax
G	S+(++i)+2	Short*	x_s + 2i +6	leaq 6 (%rdx, %rcx, 2), %rax
H	*(S+(i++)+2)	Short	M[X_s + 2i +4]	movw 4 (%rdx, %rcx, 2), %ax
I	*S--	Short*	x_s	leaq (%rdx), %rax
J	*(S--)	Short	M[x_s]	movw (%rdx), %ax