1.

	Offset				Total size	Alignment requirement
	A	B	C	D		
P1	0	4	8	12	16	4
P2	0	8	12	16	24	8
P3	0	2	----	--------	12	2
P4	0	32	----	--------	40	8x
P5	0	24	----	--------	64	8

2.

		No stack protector	Stack protector
Gcc flag			
len	Assembly for allocating stack	Sub $0x16, %rsp	Sub $0x16, %rsp
	Stack size in decimal	16	16
	Assembly for freeing stack	leave	leave
Iptoa	Assembly for allocating stack	Subq $32, %rsp	Subq $32, %rsp
	Stack size in decimal	32	32
	Assembly for freeing stack	leave	leave
	“char *s” address relative to rsp after entering lptoa	-24(%rbp)	-24(%rbp)
	“long *p” address relative to rsp after entering Iptoa	-32(%rbp)	-32(%rbp)
	“val” address relative to rsp after entering lptoa	-8(%rbp)	-8(%rbp)
Longlen	Assembly for allocating stack	Subq $0x32, %rsp	Subq $0x48, %rsp
	Stack size for decimal	32	48
	Assembly for freeing stack	leave	leave
	“x” address relative to rsp after entering longlen	-24(%rbp)	-40(%rbp)

	“v” address relative to rsp after entering longlen	-8(%rbp)	-24(%rbp)
	“buf” address relative to rsp after entering longlen	-16(%rbp)	-16(%rbp)
	Canary register name	----------	%fs:40
	Canary address relative to rsp	----------	-8(%rbp)
	Canary value	----------	40
	Assembly for erasing canary value	----------	Xorl %eax, %eax
	Assembly for canary cross check	----------	%fs:28, %rcx

3. A. $s_2=s_1-(30+8*n)$ & 0xFFFFF
   n is odd ⟶ s2=s1-(24+8*n)
   n is even ⟶ s2=s1-(16+8*n)

   B. p=(15+s2)&0xFFFFFFF0 where p is the least multiple of 16 which is greater than s2

   C. Min val of e1=1 ⟶ n=even ⟶ s1 is n%16==1
   Max val of e1=24 ⟶ n=odd ⟶ s1 is n%16==0

   D. even ⟶ s2 = 8n+16
   odd ⟶ s2 = 8n+24
   p must be aligned by 16 where s2 is the least multiple of 16 that preserve 8n size space.