NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Final Exam (Part 2) Winter 2017

14. [5 pts ea] Convert each C function call into ARM assembly language.

C code	ARM Assembly

`int8_t  *p8 ;`
`int32_t s32`
`int64_t a64[] ;`
`void f1(int8_t *, int32_t, int64_t []) ;`
 	
`f1(p8, s32, a64) ;`	

`uint64_t *p64, *f2(void) ;`
 	
`p64 = f2() ;`	

`float   average, reals[10] ;`
`void f3(float, float []) ;`
 	
`f3(average, reals) ;`	

`float *presult ;`
`float f4(void) ;`
 	
`*presult = f4() ;`	

NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Final Exam (Part 2) Winter 2017

15. [5 pts ea] Convert each C function definition into ARM assembly language.

C code	ARM Assembly

`int64_t f5(int32_t s32)`
`  {`
`  return s32 * s32 ;`
`  }`	

`int32_t f6(float real)`
`  {`
`  return (int32_t) real ;`
`  }`	

`uint64_t f7(uint64_t u64)`
`  {`
`  return u64 + u64 ;`
`  }`	

`uint64_t f8(uint64_t u64)`
`  {`
`  return u64 >> 5 ;`
`  }`	

NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Final Exam (Part 2) Winter 2017

16. [5 pts ea] Convert each C function definition into ARM assembly language.

`int32_t f9(int32_t x, int32_t a, int32_t b)`
`  {`
`  return (x < a || x > b) ;`
`  }`
	Do NOT use an IT block

`int32_t f10(uint32_t score)`
`  {`
`  return (score >= 60 && score <= 100) ;`
`  }`
	Do NOT use an IT block

NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Final Exam (Part 2) Winter 2017

17. [5 pts ea] Convert each C function definition into ARM assembly language.

	Use an IT block

`int32_t f11(uint64_t a64, uint64_t b64)`
`  {`
`  return (a64 <= b64) ;`
`  }`	

`int32_t f12(void)`
`  {`
`  int32_t f13(void), f14(void) ;`
 	
`  return f13() + f14() ;`
`  }`	

`int32_t f15(int32_t *p32, int32_t k)`
`  {`
`  return p32[k-1] ;`
`  }`	

NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Final Exam (Part 2) Winter 2017

18. [5 Pts] Write a function in ARM assembly language that calculates the single-length (32-bit) product of two 32-bit 2's complement signed operands. Unlike regular multiplication, the function should limit the product to no less than full-scale negative (0x80000000) and no more than full-scale positive (0x7FFFFFFF). The function prototype should be:

```
int32_t Product(int32_t a, int32_t b) ;
```

19. [5 pts] Write a C inline function that rotates its 32-bit unsigned argument left by 1 bit and returns the result. Use an extended asm statement inside the inline function so that the compiler is allowed to choose all of the registers.