

Student ID _____

Name _____

cs5f ____

Signature _____

CSE 5A
Final
Fall 2006

Page 1 _____ (18 points)

Page 2 _____ (26 points)

Page 3 _____ (28 points)

Page 4 _____ (16 points)

Page 5 _____ (40 points)

Page 6 _____ (44 points)

Total _____ (172 points = 164 points + 8 points EC)

This exam is to be taken **by yourself** with only your 2-sided notes, no electronic devices.

Operator Precedence Table

Operators					Associativity
- (unary)	++	--	!		right to left
*	/	%			left to right
+	-				left to right
<	<=	>	>=		left to right
==	!=				left to right
&&					left to right
					left to right
=	+=	-=	*=	/=	right to left

1. Using the operator precedence table above, evaluate each expression and state what gets printed.

```
int x = 3;
int a = 16;
int b = 11;
```

```
x = b + x - b * x / a;
printf( "%d\n", x );
```

(3 pts)

```
int x = 3;
int a = 16;
int b = 11;
```

```
x = a + b % x * x - b;
printf( "%d\n", x );
```

(3 pts)

2. What gets printed in the following blocks of statements?

```
int a = 4;
int b = 5;
int c = -7;
```

```
if ( (c < b) && !(a < b) || !(c == a) )
    printf( "True" );
else
    printf( "False" );
```

(3 pts)

```
int x = -3;
int y = 10;
int z = x + 9;
```

```
if ( (z == y) || (x < y) && !(y <= z) )
    printf( "True" );
else
    printf( "False" );
```

(3 pts)

3. Fill in the blanks for the appropriate compilation sequence. (6 pts)

- | | |
|-------------------|--------------------------|
| A) C Compiler | D) Executable Program |
| B) C Preprocessor | E) Linker/Linkage Editor |
| C) C Source Code | F) Assembler |

_____ -> _____ -> _____ -> _____ -> _____ -> _____

4. Which of the following are not valid C identifiers? **Circle** the incorrect identifiers. (16 pts)
[+1 if correct; -1 if incorrect]

for	1stOne	Nine_2_Five
Big1	Nine_To_Five	Nine-To-Five

Fill in the blanks with the appropriate types and format specifiers to output the values correctly.

```
int
main( void )
{
    _____ a = '2';
    _____ b = 2;
    _____ c_____ = "CSE 5A";
    _____ d = 4.20;

    printf( "b = %____\nd = %____\nc = %____\na = %____\n", b, d, c, a );
    printf( "c[%____] = \'%____\'\n", b, c[b] );

    return 0;
}
```

What does that last printf() statement output?

5. Write a function called checkRange that checks if the first argument is between the second and third arguments exclusive. You can assume the second argument is less than or equal to the third argument. Return the integer value 1 to indicate YES (the first argument is between the second and third argument); return the integer value 0 to indicate NO (the first argument is not between the second and third argument).

Fill in the blanks to complete this function. (10 pts)

Examples:	checkRange(10, 10, 20) would return 0	checkRange(8, 10, 20) would return 0
	checkRange(30, 20, 30) would return 0	checkRange(43, -9, 33) would return 0
	checkRange(25, 22, 44) would return 1	checkRange(19, 19, 19) would return 0

```
_____ checkRange( int value, int minValue, int maxValue )
{
    if ( _____ || _____ )
        return 0;

    else
        _____ ;
}
```

6. Write an equivalent **while** loop for the following **for** loop. (12 pts)

Equivalent **while**

```
int num, result; // Do not change
```

```
int num, result; // Same as the for loop
```

```
for ( num = 7; num < 100; num = num * 3 )
{
    result = foo( num );
    printf( "%d %d\n", result, num );
}
```

7. What gets printed in the following block of statements? (8 pts)

```
#define SIZE 8

int i;
int array[SIZE] = { -11, 2, 14, 4, 12, 3, 24, 10 };

for ( i = 0; i < SIZE; ++i )
    if ( (array[i] * 2) < 20 )
        printf( "%d\n", array[i] );
```

8. What gets printed? (8 pts)

```
#include <stdio.h>

int function1( double param1, int param2 );

void
main( void )
{
    double i = 6.69;
    int j = 5;

    j = function1( i, j );
    printf( "%d\n", j );

    return 0;
}

int
function1( double param1, int param2 )
{
    int i;

    for ( i = 8; i > param2; --i )
        printf( "%.2f\n", param1 + i );
    return (param2 + i);
}
```

9. What gets printed? (16 pts)

```
#include <stdio.h>

#define SIZE 8

char notJenny( int x );

int
main( void )
{
    char answer[SIZE];
    int i;

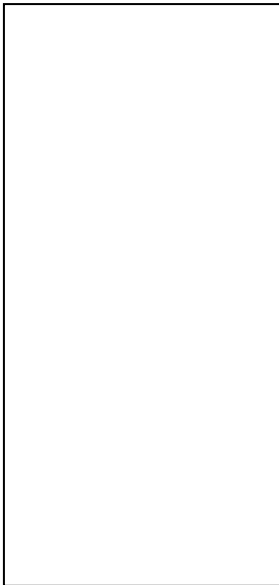
    for ( i = 0; i < SIZE; ++i )
    {
        answer[i] = notJenny( i );
    }

    for ( i = 0; i < SIZE; ++i )
    {
        printf( "%c\n", answer[i] );
    }

    return 0;
}

char
notJenny( int x )
{
    char str[SIZE] = { 'f', 'a', 'g', 'r', 'p', 'o', 'u', 'e' };

    if ( (x % 2) != 1 )
        return ( str[(x + 3) % SIZE] );
    else
        return ( str[(x + 5) % SIZE] );
}
```



10. Consider the following program. Identify the marked parts, lifetime, and scope/visibility with the corresponding letter/digit from the lists below. (40 pts)

C/C++ Program Part

- A) C Preprocessor Directive
- B) External Static Variable
- C) Function Prototype
- D) Function Definition
- E) (Formal) Parameter
- F) Internal Static Variable
- G) Local Variable
- H) Global Variable

Lifetime

- 1) During foo() call
- 2) Entire Program
- 3) During func2() call

Scope/Visibility

- WW) Within func2() Only
- XX) Just This Source Module
- YY) Within foo() Only
- ZZ) Entire Program

	<u>C/C++ Program Part</u>	<u>Lifetime</u>	<u>Scope/Visibility</u>
#include <stdio.h>	_____		
#define SIZE 17	_____		
int func2(char array[]);	_____ (entire line)		
static long johns;	_____	_____	_____
double ch = 4.20;	_____	_____	_____
void foo(char ch)	(foo(){...}) _____	_____	_____
{	(ch) _____	_____	_____
int actor;	_____	_____	_____
static int answer;	_____	_____	_____
/* Other code here */	_____	_____	_____
}			
static int func2(char fubar[])	(func2(){...}) _____	_____	_____
{	(fubar) _____	_____	_____
int result = 19;	_____	_____	_____
static char johns = 'A';	_____	_____	_____
/* Other code here */	_____	_____	_____
}			

How many times is the variable **result** in **func2()** initialized to 19 if **func2()** is called 6 times? _____ time(s)

What is the initial value of the variable **answer** in **foo()**? _____

How many times is the variable **answer** in **foo()** given this value if **foo()** is called 6 times? _____ time(s)

What is the initial value of the variable **actor** in **foo()**? _____

How many times is the variable **johns** in **func2()** initialized if **func2()** is called 6 times? _____ time(s)

Code in **foo()** that refers to the symbol/name **johns** refers to which symbol/name (state the type)?

Code in **func2()** that refers to the symbol/name **johns** refers to which symbol/name (state the type)?

11. Consider the following structure definition and variable declarations. (20 pts)

```
struct Almost_Done
{
    int a[5];
    float b[11];
    int c;
    char d[100];
    int e;
};
struct Almost_Done var1, var2, var3;
```

Fill in the blanks to complete the following tasks:

```
/* Read the value typed at the keyboard into the struct member e in var1 */
scanf( "%____\n", _____ );

/* Print all elements of struct member b in var2 EXCEPT the first and last elements */
for ( i = _____ ; i < _____ ; _____ )
    printf( "%____\n", _____ );

/* Put the answer to what is my favorite beer in struct member d in var3 */
str_____( _____ , _____ );
↑
(complete this function)
```

12. Consider the following strings variable definitions. (24 pts)

```
char s1[] = " of Suburbia";
char s2[] = "Jesus";
char s3[40];
char s4[20] = "Green Bay";
char s5[] = "-Land!";

strcpy( s3, s2 );
strcat( s3, s1 );
```

What gets printed?

```
printf( "%d", sizeof( s1 ) ); _____
printf( "%d", strlen( s3 ) ); _____
```

Fill in the blanks to complete the following tasks:

```
/* Change the 'B' in s4 to 'D' without using an explicit 'D' or 'd' */
_____ = _____ ; /* CANNOT use 'D' or 'd' */

/* Output "Green Day - Jesus of Suburbia!" in a single printf() statement. */
printf( "%__%__%__%__", _____, _____, _____, _____ );
      ↑      ↑
      (space)
```

Scratch Paper

Scratch Paper