

Student ID \_\_\_\_\_

# CSE 5A

Name \_\_\_\_\_

Signature \_\_\_\_\_

## Final Fall 2003

cs5a \_\_\_\_\_

Page 1 (12) \_\_\_\_\_

Page 2 (33) \_\_\_\_\_

Page 3 (33) \_\_\_\_\_

Page 4 (27) \_\_\_\_\_

Page 5 (40) \_\_\_\_\_

Page 6 (38) \_\_\_\_\_

This exam is to be taken **by yourself** with closed books, closed notes, no calculators.

Total (174 + 9 EC = 183)

**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;
int a = 10;
int b = 15;
```

```
x = a + b % 2 * 3 - b;
printf( "%d\n", x );
```

(3 pts)

```
int x;
int a = 10;
int b = 15;
```

```
x = b + 3 / 2 * 4 - a;
printf( "%d\n", x );
```

(3 pts)

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

```
int a = 8;
int b = 5;
int c = -7;

if ( (a < 6) || (b >= 8) || (c == a) )
    printf( "True" );
else
    printf( "False" );
```

(3 pts)

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

if ( (z != 8) && (x > y) && (z < x) )
    printf( "True" );
else
    printf( "False" );
```

(3 pts)

3. Which of the following are valid C identifiers? (Circle your answer(s).) (6 pts)

91X

FM94\_9

\_RadioHead

Bagels4Me

Darling.Nikki

Jane's\_Addiction

4. Fill in the blanks for the appropriate compilation sequence. (12 pts)

- A) Executable Program
- B) Linker/Linkage Editor
- C) C Source Code

- D) Assembler
- E) C Preprocessor
- F) C Compiler

\_\_\_\_\_ → \_\_\_\_\_ → \_\_\_\_\_ → \_\_\_\_\_ → \_\_\_\_\_ → \_\_\_\_\_

5. What gets printed? (15 pts)

```
void
main( void )
{
    int num = 1;

    switch ( num + 6 )
    {
        case 1:
            printf( "A\n" );
            num = num + 2;

        case 4:
            printf( "B\n" );
            num = num + 4;
            break;

        case 7:
            printf( "C\n" );
            num = num + num;

        case 6:
            printf( "D\n" );
            num = num + 5;
            break;

        default:
            printf( "E\n" );
            num = num + 8;
            break;
    }

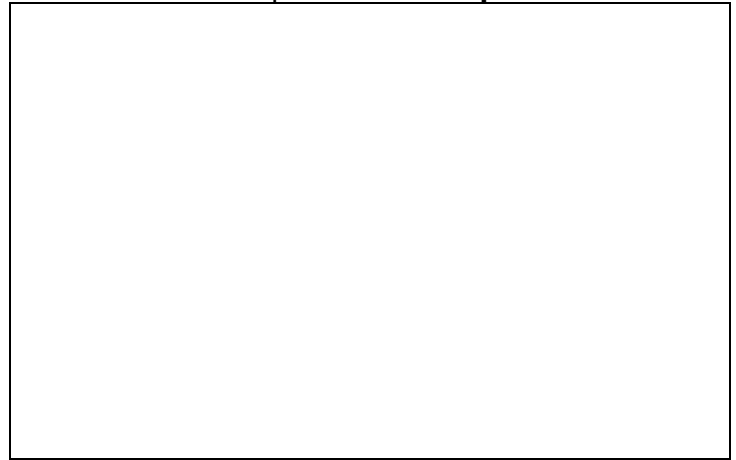
    printf( "num = %d\n", num );
}
```

What would get printed if the switch statement read  
switch( num + 2 ) instead of switch( num + 6 )?

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

```
index = 4;
while ( index > j )
{
    printf( "%d %d\n", j, index );
    --index;
}
```

Equivalent for loop

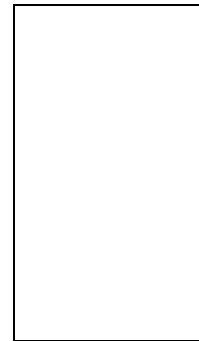


7. What gets printed in the following block of statements? (12 pts)

```
#define SIZE 8

int i;
int array[SIZE] = { 11, 7, 3, 4, 6, 2, -1, 5 };

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



8. What gets printed? (9 pts)

```
#include <stdio.h>

int function1( int var1, char var2 );

void
main( void )
{
    int i = 2;
    char j = '4';

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

int
function1( int var1, char var2 )
{
    int i;

    for ( i = 0; i < var1; ++i )
        printf( "%c\n", var2 );
    return i;
}
```



## 9. What gets printed? (27 pts)

```
#include <stdio.h>

#define SIZE 7

int function2( int array[], int size );

void
main( void )
{
    int array[SIZE] = { 4, 1, 0, 2, 3, -2, 6 };
    int i, result;

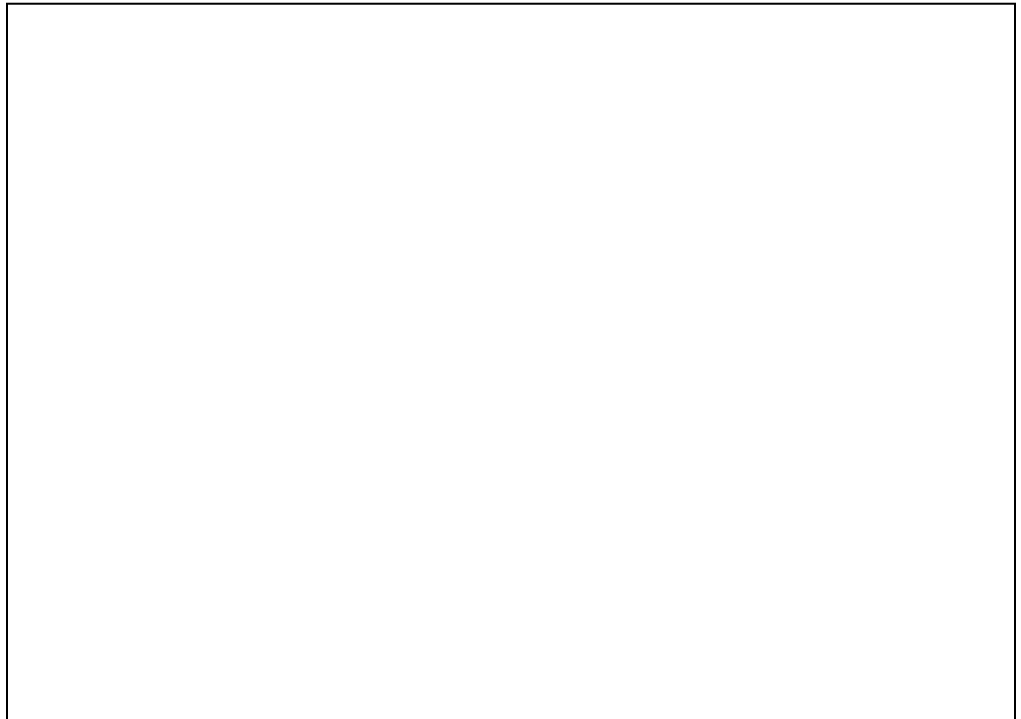
    result = function2( array, SIZE );
    printf( "Returned value = %d\n", result );

    printf( "Array elements:\n" );
    for ( i = 0; i < SIZE; ++i )
        printf( "%d\n", array[i] );
}

int
function2( int array[], int size )
{
    int i;
    int count = 0;

    for ( i = 0; i < size; ++i )
    {
        if ( array[i] >= 2 )
        {
            array[i] = 2 * array[i];
            ++count;
        }
    }

    return count;
}
```



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

**Lifetime**

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

**Scope/Visibility**

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

<u>C/C++ Program Part</u>	<u>Lifetime</u>	<u>Scope/Visibility</u>
#include <stdio.h>	_____	
#define SIZE 5	_____	
int func2( char array[] );	_____ (entire line)	
static long johns = 420;	_____	_____
double header;	_____	_____
void foo( char ch ) (foo(){...})	_____	_____
{ (ch)	_____	_____
int array[SIZE] = { 0, 3, 5, 2, 6 };	_____	_____
int result = 4; /* Other code here */	_____	_____
}		
static int func2( char johns[] ) (func2(){...})	_____	_____
{ (johns)	_____	_____
static int i;	_____	_____
int result; /* Other code here */	_____	_____
}		

How many times is the variable **result** in **foo()** initialized to 4 if **foo()** is called 8 times? \_\_\_\_\_ times

What is the initial value of the variable **i** in **func2()**? \_\_\_\_\_

How many times is the variable **i** in **func2()** given this value if **func2()** is called 8 times? \_\_\_\_\_ times

What is the initial value of the variable **result** in **func2()**? \_\_\_\_\_

How many times is the variable **array** in **foo()** initialized if **foo()** is called 8 times? \_\_\_\_\_ times

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

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

**11. Consider the following structure definition and variable declarations. (16 pts)**

```
struct Almost_Done
{
    float a;
    int b;
    char c;
    float d[8];
    int e;
};

struct Almost_Done var1, var2, var3;
```

Fill in the blanks to complete the following tasks:

```
/* Print the value of the struct member c in var3 */
printf( "%____\n", _____ );

/* Print all the elements of the struct member d in var1 */
for ( i = _____ ; i < _____ ; _____ )
    printf( "%____\n", _____ );

/* Assign the value 420 to the struct member e in var2 */
_____ = 420;
```

**12. Consider the following strings variable definitions. (22 pts)**

```
char s1[] = "Hello";
char s2[] = "Yellow";
char s3[20];

strcpy( s3, "Mellow bellow" );
```

What gets printed?

```
printf( "%d", strlen( s2 ) ); _____
```

Fill in the blanks to complete the following tasks:

```
/* Change s1 to be "Jello" instead of "Hello" */
_____ = _____ ;

/* Change the 'b' in "bellow" to 'B' in s3 using the toupper() function */
_____ = _____ ;

/* Output "Jello Mellow Bellow Yellow" in a single printf() statement. */
printf( "%__ %__ %__\n", _____, _____, _____ );
```

## Scratch Paper