CSCI 3133.01 MIDTERM FALL 1999

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Grade\_\_\_\_\_\_\_\_

1. For the loop below:

a. How many lines are printed? \_\_\_\_\_\_3\_\_\_\_\_\_

b. What value does i have at termination? \_\_\_\_\_\_3\_\_\_\_\_\_

c. Change the **for** loop into a **while** loop.

for(i = 0; i <=2; ++i) int I = 0;

printf("i=%d\n",i); while( I <= 2 )

{

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

I++;

}

2. Write 6 lines of C to print "i is negative", "i is zero" or

"i is positive", according to the value of i. Avoid any useless

or redundent tests.

if( i < 0 )

printf( "I is negative" );

else if( i == 0 )

printf( "I is zero" );

else

printf( "I is positive" );

3. Assuming a = 1; b = 2; c = 3; d = 4; all int evaluate

a. a + b \* c \_\_\_\_\_\_\_7\_\_\_\_\_\_

b. -b \* -c \_\_\_\_\_\_\_6\_\_\_\_\_\_

c. a + c < b + d \_\_\_\_\_\_\_1\_\_\_\_\_\_

d. a < d && c < b \_\_\_\_\_\_\_0\_\_\_\_\_\_

e. c/d < a \_\_\_\_\_\_\_1\_\_\_\_\_\_

f. d %b != b/c \_\_\_\_\_\_\_1\_\_\_\_\_\_

g. a < b || d < c \_\_\_\_\_\_\_1\_\_\_\_\_\_

4. Write the current values of a, b and c after executing each loop: **You won’t be able to do this. Don’t worry!**

main( ) a[0] a[1] a[2] a[3] a[4]

{

short a[5];

for(i = 0; i < 5; ++i)

a[i] = i \*2; 0\_\_ \_\_2\_\_\_ \_\_\_4\_\_ \_\_6\_\_\_ \_\_8\_\_\_

for(i =0; a[i] !=2; ++i)

a[i] = 9; 9\_\_ \_\_2\_\_\_ \_\_\_4\_\_ \_\_6\_\_\_ \_\_8\_\_\_

while(a[++i] != 6)

a[i] = 7; ; 9\_\_ \_\_2\_\_\_ \_\_\_7\_\_ \_\_6\_\_\_ \_\_8\_\_\_

do

{

a[i] = 5;

}while(i++<3); ; 9\_\_ \_\_2\_\_\_ \_\_\_7\_\_ \_\_5\_\_\_ \_\_8\_\_\_

for( ; i>0; --i)

switch (a[i]%3)

{

case 0:

--a[i];

break;

case 1:

++a[i];

break;

case 2:

a[i] -= 2;

break;

}

9\_\_ \_\_0\_\_\_ \_\_8\_\_\_ \_\_\_3\_\_ \_\_\_6\_\_

}

5. What is the output of each of the following programs?

/\* loop 1 \*/ /\* loop 2 \*/ /\* loop 3 \*/

#include <stdio.h> #include <stdio.h> #include <stdio.h>

main ( ) main ( ) main ( )

{ { {

int i; int i; int i;

i = 5; i = 5; for(i = 5; i < 5;++i)

do while (i < 5) printf("%d\n",i);

{ { printf("%d\n",i);

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

++i; ++i;

} while (i < 5) }

printf("%d\n",i); printf("%d\n",i);

} }

OUTPUT OUTPUT OUTPUT

5 5 5

6

6. What is the exact order of the output for trees?

/\* trees - print tree name \*/

main ( )

{

printf("6 trees:\n");

t1();

t4();

}

t1()

{

printf("ash\n");

t2();

}

t2()

{

t3();

printf("fir\n");

}

t3()

{

printf("beech\n");

}

t4()

{

printf("dogwood\n");

t5();

printf("yew\n");

}

t5()

{

printf("gum\n");

}

6 trees:

ash

beech

fir

dogwood

gum

yew

7. Cicle the ones that are not identifiers.

X 3id

\_yes

o\_no\_o\_no

X 00\_go

X start\*it

X one\_I\_won't

X me\_to-you

xYshouldI

X int

8. Write the values of a, b and c after each print statement.

a b c

int a, b=0, c=0;

a = ++b + ++c;

printf( "%d %d %d\n", a, b, c ); \_2\_\_ \_1\_\_ \_1\_\_

a = b++ + c++;

printf( "%d %d %d\n", a, b, c ); \_2\_\_ \_2\_\_ \_2\_\_

a = ++b + c++;

printf( "%d %d %d\n", a, b, c ); \_5\_\_ \_3\_\_ \_3\_\_

a = b-- + --c;

printf( "%d %d %d\n", a, b, c ); \_5\_\_ \_2\_\_ \_2\_\_

9. Write a statement that will open the file *infile.dat* for reading and another statement that will open the file *outfile.dat* for writing.

fopen( "infile.dat", "r" );

fopen( "outfile.dat", "w" );

10. Write a function **double power( double x, int n )** that will compute xn, the *n*th power of *x*.

double power( double x, int n )

{

double value = 1;

if( n == 0 )

value = 1;

else

for( int k= 1; k<n; k++ )

value = value \* x ;

return value;

}

11. What gets printed?

int a = 1, b = 2, c = 3;

a += b += ++c;

printf( "%5d%5d%5d\n", a, b, c );

{

float b = 4.0;

int c;

a += c = 5 \* b;

printf( %5d%5.1f%5d\n", a, b, c );

}

printf( "%5d%5d%5d\n", a, b, c );

7 6 4

27 4.0 20

27 6 4

12. Write a recursive function that computes the following: **You won’t be able to do this. Don’t worry!**

s(n) = 2 + 4 + 6 + … + 2n where n >=1

int S( int n )

{

if( n == 1 )

return 2;

else

return ( S(n-1) + 2\*n );

}

13. a) (True/False) A C program must have a function called **main**.

b) (True/False) An invoking function must pass arguments to the invoked

function.

c) (True/False) Every function returns a value to its invoker.

d) (True/False) A function may contain more than one **return** statement.

e) (True/False) A function may return more than one value at a time.

14. View the following code segments. If there are any syntax errors, explain.

1. void func1( int param1, float param2 );

{

…

}

Doesn't have a semicolan at the end.

1. void func2( int, float )

{

…

}

No variable names

1. int status( code char, time float );

Variable names and types interchanged

1. return value1, value2;

Can only have one return value.

15. a) Write a #include directive to include the system file *time.h*

#include <time.h>

b) Write a #include directive to include the user file *my\_definitions.h*

#include "my\_definitions.h"

16. Suppose there is an array **sample1** defined as

float **sample1**[ 10 ];

Explain what is passed into the following functions. **You won’t be able to do this. Don’t worry!**

1. funct1( sample1 );

Sends the address of the array ( or the address of the first array element )

1. funct2( sample1[ 2 ] );

The value stored in the third array element

17. Given the definition

double arr4[ 20 ][ 4 ][ 6 ][ 2 ]

write **You won’t be able to do this. Don’t worry!**

1. A call to a function **fun1**, with the array arr4 as the only argument

fun1( arr4 );

1. **fun1**'s header

int fun1( double arr4[][4][6][2]

19. Write a function **check** with two parameters, **str** and **c**. The parameters **str** is a **char** array, and the parameter **c** is a **char**. The function **check** returns 1 if **c** is in **str** and 0 otherwise. **You won’t be able to do this particular question because you have not learnt arrays. But, you should be able to do something similar to this.**

int check( char str[], char c )

{

if( strchr(str, c ) != NULL

return 1;

else

return 0;

}