CS211

Sample Exam Problems

1. Look at the program below.

```
#include <iostream.h>

 void f( int i )
    {
    if( i = 4 || i = 5 )   return;
    cout << "hello world\n" ;
    }

 int main()
   {
   f( 3 );
   f( 4 );
   f( 5 );
   return 0;
   }
```

What exactly will be printed?

2. Here are three definitions:

a. Def: A year y is a **century year** if y is divisible by 100.
Example: y=300 is a century year, but y=150 is not.

b. Def: A year y is a **non-century year** if y is not a century year.
Example: y=150 is a non century year. Y=4 is a non-century year.

c. Def: a year y is a **leap year** if it is a non-century year that is divisible by 4, or a century year that is divisible by 400. Nothing else is a leap year.
Example: y=4 is a leap year, y=100 is not a leap year, y=400 is a leap year.

Question: a Write a function

bool leapyear(int y)

that when passed a year y will return ***true*** if y is a leap year and ***false*** if not.

3. Write a function

bool ok(int q[])

which takes an array q and returns *true* if the array represents a valid configuration of the eight queens and returns *false* otherwise.

4. Write a function

bool ok(int q[])

for the "Eight numbers in a cross" problem. Note that the helper array int a[8][5] should be local to the function (and static) since that is the only place that it is used.

5. In the "Eight numbers in a cross" problem, if you are given the labeling if the "cross" how do you construct the helper array a.

6. What will the following code print?

Note: the sizeof() function returns the number of bytes of the argument passed to it.

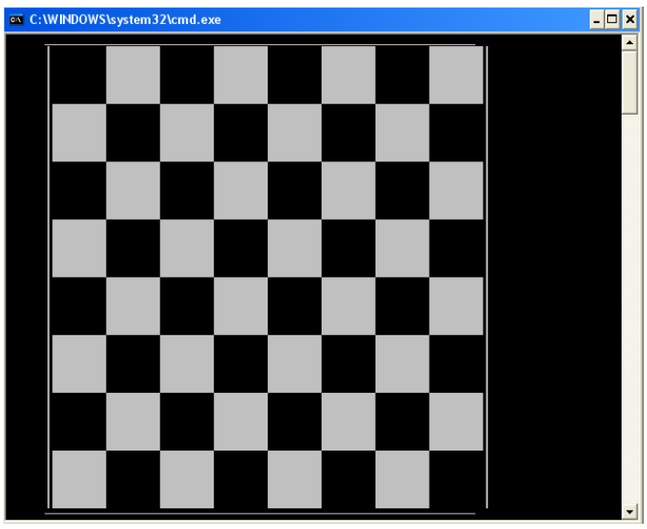
```
#include <iostream>
using namespace std;

int main( )

{ typedef  int my_2darray[1][1];

my_2darray  b[3][2];

cout<<sizeof(b)<<endl;

cout<<sizeof(b+0)<<endl;

cout<<sizeof(*(b+0))<<endl;

// the next line prints 0012FF4C

cout<<"The address of b is: "<<b<<endl;

cout<<"The address of b+1 is: "<<b+1<<endl;

cout<<"*(b+1) is: "<<*(b+1)<<endl<<endl;

cout<<"The address of &b is: "<<&b<<endl;

cout<<"The address of &b+1 is: "<<&b+1<<endl<<endl;


return 0;
}
```

7. We looked at the following code to print a chessboard:

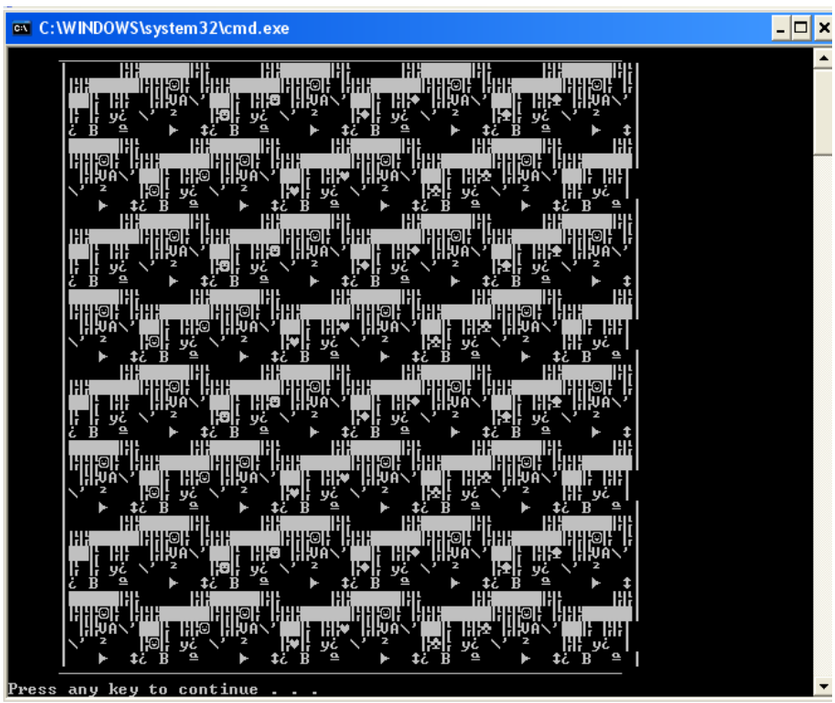
```
// Print a chessboard
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{       int i,j,k,l;

    typedef char box[5][7];

    box bb,wb,*board[8][8];

    //fill in bb=black box and wb=whitebox
    for(i=0;i<5;i++)
        for( j=0;j<7;j++)
        {wb[i][j]=' ';
        bb[i][j]=char(219);
        }

    //fill board with pointers to bb and wb in alternate positions
    for(i=0;i<8;i++)
        for(j=0;j<8;j++)
            if((i+j)%2==0)
                board[i][j]=&wb;
            else
                board[i][j]=&bb;

    // print the board via the pointers in array board

    // first print upper border
    cout<<"    ";
    for(i=0;i<7*8;i++)
        cout<<'_';
    cout<<endl;

    // now print the board
    for(i=0;i<8;i++)
        for(k=0;k<5;k++)
        {cout<<"    "<<char(179); //print left border
        for(j=0;j<8;j++)
            for(l=0;l<7;l++)
                cout<<(*board[i][j])[k][l];
        cout<<char(179)<<endl; // at end of line print bar and then newline
        }

    //before exiting print lower border
    cout<<"    ";
    for(i=0;i<7*8;i++)
        cout<<char(196);
    cout<<endl;
  system("PAUSE");
  return EXIT_SUCCESS;
}
```

a. Since the name of an array is the address of the first element of the array, will the code in yellow above: board[i][j]=&wb; work equally well if we remove the “&” and write board[i][j]=wb;? Why or why not. If the answer is no, then what will the compiler complain about?

b. When I run the code above with the code in blue like it is in the program (cout<<(*board[i][j])[k][l];) I get the chessboard correctly printed, like this:



But , if we move the “*” outside the parenthesis and write this: cout<<*(board[i][j])[k][l];, I get the following output. What **exactly** went wrong???



8. Know how to do these, to trace functions like these and to debug functions like these:

```
// recursive power , compute xn
int exp(int x, int n){
        if(n==_______)
                return _________;
        return _____________*exp(_________);
}
void main(){
        int a,b;
        cin >>a>>b;
        cout<<endl<<exp(a,b)<<endl;
}
// recursive print of a string
void print(char* s){
        if(*s==_____________)
                return;
        cout<<*s;
        print(_______________);
}
void main(){
        char a[6]="hello";
        print(a);
}
// recursive reverse print of a string
void rev_print(char* s){
        if(*s==_____________)
                return;
        ___________________________________;


        ___________________________________;
}
void main(){
        char a[100];
        cin>>a;
        rev_print(a);
}
```

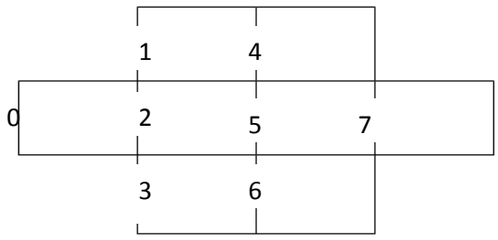
```
// recursive reverse print of an integer
void rev_print(int n){
	if(n/10==0){
		cout<<n%10;
		return;
	}
	cout<<_____________________________;
		rev_print(_____________________);
}
void main(){
	int a;
	cin>>a;
	rev_print(a);
}
// recursive print of an integer
void print(int n){
		if(n/10==0){
		cout<<n%10;
		return;
	}
	print(_________________);
	cout<<_____________________;
}
void main(){
int a;
cin>>a;
	print(a);
}
```

9. One of the programs that you wrote this semester was “Eight numbers in a cross.”

The problem was to write a program which allocates the integers 1-8 to the squares in the figure above, subject to the restrictions that no two adjacent squares contain consecutive integers.

By adjacent we mean vertically, horizontally, or diagonally.

Say we **label** the eight “boxes” as follows:



In the above diagram, the numbers 0-7 in the boxes are not the values in the boxes but the “names” of the boxes.

In the program we used a “helper array:

```
int a[8][5]={
{-1},
{           },
{           },
{           },
{           },
{           },
{           },
{           },
{           }
};
```

Question:

Given the labeling of the boxes of the cross as above, fill in the values in the “helper array” a.

10. Write a function

int * return_an_array(int n)

which takes an integer n and returns a pointer to an array (allocated off of the heap in the function) containing the digits of n in the appropriate positions. If n = 0 return 0 (which is the "null" pointer).

Recall an array name is just a pointer to the first element of the array.

11. Write a function

int number_of_primes(int from, int to)

that returns the number of prime numbers between from and to.

12. Write a function

int * return_matches(int a[], int & size,TEST t)

which returns an array (allocated off of the heap in the function) containing only the elements of array a that pass the "test" t. The function iterates through all the elements of a, and constructs a new array containing all the elements that pass the test.

When the parameter corresponding to "size" is passed in (by reference), it contains the size of the array a. In the function that variable is modified, so that when "return_matches" returns, the parameter will contain the number of matches found. This will allow us to print the array returned by the function, since we know how big it is.

What is a "test"? It's a Boolean function that tests if an integer meets a specific criteria.

For example

```
bool is_even(int n){
    return n%2==0;
}
```

To make this work we could use a typedef like:

typedef bool TEST(int)

so now TEST becomes a type and the function "is_even" is indeed of type TEST since it is a function of a single integer variable and returns a bool.

13. Know how stable marriage works.

14. Write a "memoized" Fibonacci function.

15. We spent LOTS of time on the s Queens Problem in its various forms. It might be a good idea to review that material.

Good Luck!