1. (9 pts) Show what will be output by the cout's in this program. As in normal program execution, any update to a variable should affect the next statement.
   (Note: boolalpha simply causes Booleans to show on the screen as 'true' or 'false' when printed by cout).

```
#include <iostream>
#include <algorithm>
#include <cmath>

using namespace std;

int f1(int x, int y)
{
      return (double)(x/y);
}

int main()
{
      int x = 10, y=4, a = 11, b = 8;
      double z = 12.5;

      cout << 19 % 5 << endl;                          _____________________
      cout << 5 / 2 << endl;                           _____________________
      cout << 5 + 3 % 2 * x / 4 << endl;               _____________________
      cout << (a++ + b) << endl;                       _____________________
      cout << (++a - b) << endl;                       _____________________
      cout << boolalpha << (x && (y || y/x)) << endl;  _____________________
      cout << pow(2, y) << endl;                       _____________________
      cout << min(max(min(x,y),x-1),(int)z) << endl;   _____________________
      cout << f1(12,b) << endl;                        _____________________
      return 0;
}
```

2. (4 pts) Given the following declarations, fill in the table with the **type** that will result from the given expression. We do not want a value…only the **type**.

```
int x;
char word[10];
char** s;
double* p;
```

Expression	Type
&x	
word	
*word	
word[5]	
**s	
&p	
*(word+1)+1	
s + 2	

3. (1 pts) **True/False**:_____ In the code below, 3 will be printed.

```
int data[3] = {3,2,1};
cout << data << endl;
```

4. (2 pts) Given the following declarations and call to the function named **doit**, infer and write a correct **prototype** (i.e. pre-declaration of the function) for doit.

```
char c='C';
char* x[10];
double z=3.14;
bool result = doit(&c, x, &z);
```

Prototype for doit:

5. (6 pts) The following function should be written to generate random numbers in the interval [0,99] (inclusive) and return the number of iterations required for the total sum of all generated numbers to reach at least 1000. Assume proper includes and that srand() has been called already. Fill in the details.

```
int generate()
{    int total = _____, cnt = 0;

   //write the loop kind and condition
    ___________( ______________________________ )
    {
        int r = _____________________;
        total = _____________________;
        cnt++;
    }
    return _________;
}
```

6. What will be printed by the program below.

```
void mystery(int *a, int n)
{   for(int i = 0; i<n; i++)
        a[i] = a[n-i-1];
}
int main(){
    int x[] = {3,7,4,-1,5,-2,8};
    mystery(x,7);
    for(int i=0; i < 7; i++)
       cout << x[i] << " ";
    return 0;
}
```

Program output: _____

7. Using similar meaning of the image library used in class and lab, what 5x5 image does this program draw assuming an all white starting image:

```
unsigned char image[5][5];
//code to initialize image to white not shown
const int MAX = 5;

int r = 0, c = 0, delta = 1;
int *same  = &c, *change = &r;

for(int i=0; i < MAX; i++)
{
    for(int j = 0; j < MAX-i; j++)
    {
       image[r][c] = 0;
       *change += delta;
    }

    *change -= delta;

    int* temp = change;
    change = same;
    same = temp;

    if( (i % 2) == 1){
       delta = -delta;
    }

    *change += delta;
}
```

Show the resulting image using the grid below.

8. (7 pts) Fill in the missing parts for: `bool isdigits(char* str);` which should return true if all characters in the string are digits (ASCII '0'-'9')

```
bool isDigits(char* str)
{
  // no other variables may be declared
  while( ______ != ______){
    if( ! (*str >= ____) ___ (_____ <= ____) ){
      return false;
    }
    ___________;
  }
  return true;
}
```

10. (9 pts.) Show what will be printed by this program (ignore missing #includes, etc.) if it is run via the command line: `./prog1 aaab baabba`

```
bool f2(char s[], char c)
{
  int i=0;
  while(s[i] != 0){
    if(s[i] == c) {
      s[i] = '-';
      return true;
    }
    i++;
  }
  return false;
}

bool f1(char* s)
{
  while( f2(s, 'a') ){
    if( ! f2(s, 'b') ){
      return false;
    }
  }
  return !f2(s, 'b');

}

int main(int argc, char* argv[])
{
  char str1[10] = "";

  if( f1(str1) ) cout << "str1 yes" << endl;
  else cout << "str1 no" << endl;
  cout << str1 << endl;

  if( f1(argv[1]) ) cout << "argv[1] yes" << endl;
  else cout << " argv[1] no" << endl;
  cout << argv[1] << endl;

  if( f1(argv[2]) ) cout << "argv[2] yes" << endl;
  else cout << " argv[2] no" << endl;
  cout << argv[2] << endl;

  return 0;
}
```

Program output:

1. (9 pts.) Show what will be output by the cout's in this program. (Note: boolalpha simply causes Booleans to show on the screen as 'true' or 'false' when printed by cout):

```
#include <iostream>
#include <algorithm>
#include <cmath>

using namespace std;

int f1(int x, double y)
{
     return (int)(x*y);
}

int main()
{
     int x = 10, y=4;
     double z=10.5;

     cout << 15 % 7 << endl;                          _____
     cout << y/x << endl;                             _____
     cout << 5 + 3 % 2 * x / 4 << endl;               _____
     cout << (x++ + y) << endl;                       _____
     cout << (++x - y) << endl;                       _____
     cout << boolalpha << (!x && (y || y/x)) << endl; _____
     cout << pow(2, x) << endl;                       _____
     cout << max(min(max(x,y),x+y),(int)z) << endl;   _____
     cout << f1(y,z) << endl;                         _____
}
```

2. (2 pts.) a) Given the declaration: `char word[10];` What ***type*** will the expression **`word`** evaluate to? Circle your choice.

   a. `char`

   b. `int*`

   c. `char*`

   d. `char**`

   b) What **value** will the expression **`word`** evaluate to?

3. (4 pts.) Given the following declarations and call to the function named **doit**, infer and write a correct **prototype** (i.e. pre-declaration of the function) for doit.

```
char c='C'; int X[10]; double z=3.14;
bool result = doit(c, X, &z);
```

Prototype for doit:

4. (5 pts.) Examine the program below (don't worry about #includes, etc.). What variables could legally be printed (i.e. would be in scope / accessible / visible) at each of the cout statements in the program. In the table below place a **check mark or x** in the corresponding cell if the variable **IS** in scope and can be printed by the corresponding cout. Leave the cell blank if the variable is NOT in scope for the given cout.

```
int a;
void f1(char c)
{  cout << ____ << endl; // cout1
}
int main()
{
  int i = 0;
  for(;i<10;i++){
      double z = pow(y, i);
  }
  cin >> a;
  if(a < 1){
      double y = 2.0/3.0;
  }
  cout << ____ << endl;  // cout2
}
```

	a	c	y	i	z
cout1					
cout2					

5. (4 pts.) Given the following array declarations and indexed accesses, compute the address where the indexed value will be in memory. Assume the array starts at location 200 on a 64-bit computer.

   a. `int x[10]; x[5] is at: _____`

   b. `char c[10][4]; c[2][1] is at: _____`

   c. `double d[3][4][4]; d[1][2][3] is at: _____`

   d. `char *n[10]; n[3] is at: _____`

6. (2 pts.) Consider the following program fragment. The code is trying to find a random number n for which do_it() averages >= 3.5 for 10 iterations. do_it(n) always returns an integer in the range [0,5]. **Find and fix the problem by marking up the code**.

```
double average=0.0;
int n;
while(average < 3.5)
{
    int r=0; n = rand();
    for(int i=0;i<10;i++)
    {
        r += do_it(n);
    }
    average = (double)(r / 10);
}
```

7. (6 pts.) What does the following program output?

```
#include <iostream>

using namespace std;

int main()
{
      for(int i=0; i<10; (i<3?i++:i+=2) )
      {
            if( !(i%3))
            {
                  continue;
            }
            else if( i % 7 == 0)
            {
                  break;
            }
            cout << i << endl;

      }

}
```

Program Output:

8. (12 pts.) Show the output of this program by tracing the code execution. For this code you can assume the first 10 calls to rand() returns the following sequence [18, 12, 8, 17, 1, 13, 10, 3, 16, 5]

```
#include <iostream>
#include <cstdlib>

using namespace std;

void m1(int *a, int *b)
{
	int c = *b;
	*b = *a;
	*a = c;
}

void m2(int *X, int n)
{
	for(int i=0;i<n-2;i++)
	{
		int j = rand() % (n-i);
		m1( (X+i) , X+i+j);
	}
}

int main()
{
	int array[] = {0,1,2,3,4};

	srand(1234);
	m2(array, 5);

	for(int i=0;i<5;i++)
		cout << array[i] << " ";
	cout << endl;

}
```

Output:

10. (8 pts.) Billy Bruin was attempting to write a program that would read in up to 16 ones or zeros of a binary number. The program calculates the value of the number and prints it out. For example if you enter 101x it will output “101=5” He has several questions and a few bugs. Can you help him out?

```
1   #include <iostream>
2   #include <cmath>
3
4   using namespace std;
5
6   int main()
7   {
8         unsigned char str[  ], c;
9         unsigned int val=0;
10        int i;
11        for(i=0;i<16;i++)
12        {
13              cin >> c;
14              if( c != '0' || c != '1')
15              {
16                    _____________;
17              }
18              str[i] = c;
19        }
20
21        int len = i;
22        for(i=0;i<len;i++)
23        {
24              if( str[i] == '1')
25              {
26                    val += pow(2, len - i);
27              }
28        }
29        cout << str << "=" << val << endl;
30  }
```

Note: Binary works by multiplying each 1 or 0 from location x in the array by a power 2^x but unlike an array. But unlike an array, the right-most location is location x. So the right-most bit is worth 2^0, then the second bit from the right is worth 2^1, and so on.

a. Line 8: Billy wasn’t sure how large to declare the array **str**. Please complete it for him.
b. Line 16: Billy wanted to immediately quit the for loop if he received something other than a ‘1’ or ‘0’. Complete line 16.
c. After making these changes, Billy’s first for-loop never seems to do anything. Can you fix it?
d. After that fix, Billy notices that the final cout sometimes prints extra garbage characters before the equals sign. Add something at line 20 to fix this.
e. Finally, Billy notices that his code seems to be **almost** working, except all of his answers are too big by a factor of two. Find the mistake and write the correction below:

Line num: ____ Changed code:

1. Short answer. Fill in the best response (5pts).

   The name of an array is an expression. What value does that expression have?

   _____

   ~~What are the four attributes of a variable in C/C++?~~ _____

   _____

   In what order are the four pieces of a for() loop executed? _____

   _____

   If you know the type of a variable in C++ you know how much _____ has been allocated to that variable.

   "Memory address of X" is another way of saying "a _____ to X".

   ~~A C-string is not a type, it is a _____ on how to use a char array to store text.~~

   The parameters of a function definition are known as _____, while the parameters passed to a function when called are _____.

   Functions in C/C++ use pass-by-_____ for the input parameter and return value.

   In C/C++ any integer value can be considered true or false. What convention is used?

   _____

   Make the following expressions true:

   sizeof(char) == _____

   sizeof(double) == _____

   sizeof(int) == _____

2. (9 pts) Show what will be output by the cout's in this program. As in normal program execution, any update to a variable should affect the next statement.
   (Note: boolalpha simply causes Booleans to show on the screen as 'true' or 'false' when printed by cout).

```
#include <iostream>
#include <algorithm>
#include <cmath>

using namespace std;

double f1(int x, int y)
{
      return (x/y);
}

int main()
{
      int x = 10, y=4, a = 11, b = 8;
      double z = 12.5;

      cout << 19 % 5 << endl;                          _____________________
      cout << 5 / 2 << endl;                           _____________________
      cout << 5 + 3 % 2 * x / 4 << endl;               _____________________
      cout << (a++ + b) << endl;                       _____________________
      cout << (++a - b) << endl;                       _____________________
      cout << boolalpha << (x && (y || y/x)) << endl;  _____________________
      cout << pow(2, y) << endl;                       _____________________
      cout << min(max(min(x,y),x-1),(int)z) << endl;   _____________________
      cout << f1(12,b) << endl;                        _____________________
      return 0;
}
```

8. (3 pts) What will be printed by the program below.

```
void mystery(int a[], int n)
{   for(int i = 0; i<n; i++)
        a[i] = a[n-i-1];
}
int main(){
    int x[] = {3,7,4,-1,5,-2,8};
    mystery(x,7);
    for(int i=0; i < 7; i++)
       cout << x[i] << " ";
    return 0;
}
```

Program output: _____

9. (1 pts) What does the following code print out?

```
int x=1;
int y[] = {1,2,3};
bool z=true;

 cout <<  y && ( !x || ( z && !( x – 1) ) ? “Hello” : “Goodbye!” << endl;
```

Output:_____