1. Given the following C code:

```
float freezingPoint(char scale) {
        if (scale == 'F') { return 32.0; }
        if (scale == 'C') { return 0.0; }
        if (scale == 'K') { return 273.0; }
        return -1.0;
}
```

Identify the:

a) Return Type: float

b) Function Name: freezingPoint

c) Arguments: char scale

d) Embodiment:

```
if (scale == 'F') { return 32.0; }
if (scale == 'C') { return 0.0; }
if (scale == 'K') { return 273.0; }
return -1.0;
```

2. Given the following C code:

```
#include <stdio.h>
int add(int x,int y) { return x+y; }
int sub(int x,int y) { return x-y; }

int main() {
      int a=3; int b=7; int c;
      c = sub(add(a,b),sub(b,a));
      printf("c is %d\n",c);
      return 0;

}
```

What will get printed? "c is 6"

c=sub(add(3,7),sub(7,3))=sub(10,4)=6

3. Given the following C code:

```
#include <stdio.h>
int main(int argc, char **argv) {
      int i;
      for (i=0; i<argc; i++) {
            printf("argv[%d]=%s\n",i,argv[i]);
      }
      return 0;
}
```

What will get printed if this is compiled into an executable file called "printArgs", and you run the command "./printArgs a b c"?

argv[0]=./printArgs
argv[1]=a
argv[2]=b
argv[3]=c

4. Write a function called "quadratic" that takes three int arguments (call them "a", "b", and "c"), and one float argument called "x", and returns the float value of (a multiplied by x multiplied by x) plus (b multiplied by x) plus c. Test your function with the following "main" function:

```
int main(int argc, char **argv) {
      printf("quadratic(1,2,1,3.0)=%f\n",quadratic(1,2,1,3.0));
      printf("quadratic(1,2,1,4.0)=%f\n",quadratic(1,2,1,4.0));
      printf("quadratic(1,-2,6,3.0)=%f\n",quadratic(1,-2,6,3.0));
      return 0;
}
```

```
float quadratic(int a, int b, int c, float x) {
        return a*x*x + b*x + c;
}
```

```
laptop:~/CS211/hw/hw02>quad
quadratic(1,2,1,3.0)=16.000000
quadratic(1,2,1,4.0)=25.000000
quadratic(1,-2,6,3.0)=9.000000
```