C Pointers Worksheet

1. Declare a pointer to a short int and a pointer to a float.

2. Of what use is the sizeof() operator?

3. In a given operating system, a pointer to a short int is 32 bits wide. How wide is a pointer to a long int in this same system?

4. Assume p is a pointer to a float. Further, assume, the value of p is 1000 (i.e., the address of the float it points to is 1000). The value of the float is 17.6. What value is p++? Define in words what *p and &p mean. Is there a way to determine the values of *p and &p given the info above?

5. Given the initializations and memory map at the top, fill out the memory map on the bottom after the code has executed. Assume pointers are 32 bits wide.

```
long int x=100;
long int *y;
long int **z;

top   ---------------                code
      |             |
      |             |                y=&x;
      |             |                z=&y;
      |             |                x++;
      |             |                *y=*y++;
      |             |                *z=*z++;
      ---------------                z++;
8000  |   100       | x
      ---------------
      |             |
      ---------------
5000  |             | y
      ---------------
      |             |
      ---------------
2000  |             | z
      ---------------
      |             |
      ---------------
```

```
     map after code executes

top  ---------------
     |             |
     |             |
     |             |
     |             |
     |             |
     |             |
     ---------------
8000 |             | x
     ---------------
     |             |
     ---------------
5000 |             | y
     ---------------
     |             |
     ---------------
2000 |             | z
     ---------------
     |             |
     ---------------
```

C Pointers Worksheet Answers

1. Declare a pointer to a short int and a pointer to a float.

```
short int *a;
float *b;
```

2. Of what use is the sizeof() operator?

This operator returns the size in bytes of its argument. It can be used to find the size of pointers or aggregate data such as structures.

3. In a given operating system, a pointer to a short int is 32 bits wide. How wide is a pointer to a long int in this same system?

All pointers in a system are the same size regardless of what they point to, so 32 bits.

4. Assume p is a pointer to a float. Further, assume, the value of p is 1000 (i.e., the address of the float it points to is 1000). The value of the float is 17.6. What value is p++? Define in words what *p and &p mean. Is there a way to determine the values of *p and &p given the info above?

p++ is 1004, the address of the “next float” because floats are 4 bytes in size.

*p means the value at the location given by p, or the value at location 1000 in this case (the value of the float).

&p is the address of the pointer itself (where the pointer p resides in memory).

5. Given the initializations and memory map at the top, fill out the memory map on the bottom after the code has executed. Assume pointers are 32 bits wide.

```
top  ---------------                  code
     |             |
     |             |                  y=&x;
     |             |                  z=&y;
     |             |                  x++;
     |             |
     |             |                  *y=*y++;
     ---------------                  *z=*z++;
8000 |   101       | x                z++;
     ---------------
     |             |
     ---------------
5000 |   8000      | y
     ---------------
     |             |
     ---------------             This is what you have after
2000 |   5000      | z           the first 3 lines of code.
     ---------------
     |             |
     ---------------

     map after all code executes

top  ---------------
     |             |
     |             |
     |             |
     |             |
     |             |
     |             |
     ---------------
8000 |   101       | x
     ---------------
     |             |
     ---------------
5000 |   8004      | y
     ---------------
     |             |
     ---------------
2000 |   5008      | z
     ---------------
     |             |
     ---------------
```