ECE 222 System Programming Concepts
Lectures 4 – Arrays, strings

Surprisingly, I found that this was all review, whereas the lecture 2 stuff was not. Still, many students were not completely clear on this yet, so it was good to go through.

What is an array?

An array is a block of memory divided into cells of the given size. The cell size is determined by the basic data type, as just presented (char, int, float, double). The number of cells is given in the variable declaration. Examples:

```
int a[10];          /* 10 cells, each cell 4 bytes (32 bits) */
float b[7];         /* 7 cells, each cell 4 bytes (32 bits) */
double c[3];        /* 3 cells, each cell 8 bytes (64 bits) */
char d[12];         /* 12 cells, each cell 1 bytes (8 bits) */
```

Cells are accessed using indices, counting from zero. Examples:

> [I run through these examples live, to demonstrate a seg fault. It won't happen until you go pretty far off the array bounds. Ask class why.]

```
a[2]=5;
b[3]=4.0;           /* good habit – use decimal */
c[3]=14.7;          /* index past bounds – what happens? */
d[0]='a';           /* what is value of d[0] now? */
```

A computer can only store ones and zeroes! There is no such thing as an 'A' or ';' in a computer. The ASCII code system is a standard for relating common symbols, like those, to a number range 0-255. If a program knows it is supposed to interpret the numbers using ASCII, it knows that a value of 65 means the A symbol, for example.

Examples (run these live):

```
printf("%d\n",d[0]);
printf("%c\n",d[0]);
printf("%d\n",'b');
printf("%c\n",'b');
```

To see the whole ASCII table, try:

```
for (i=0; i<255; i++)
  printf("%d %c\n",i,i);
```

What is a string?

A string is just a special type of array. It is an array of char, ended by the NULL character. Example:

```
char d[12];
d[0]='H'; d[1]='e'; d[2]='l'; d[3]='l'; d[4]='o';
d[5]='\0';        /* \ codes are used for non-printable symbols */
```

The value of the NULL character is zero. NULL is simply another way of saying zero. It is used to specify a specific “type” of zero. For example, you might want to use a different zero for a whole number (0) than for a real number (0.0) to show that they are telling you slightly different information. NULL is a way to say zero for a char representing an ASCII symbol that means end of string. (NULL can also mean zero for an address, but we’ll get to that later in the semester.) Bottom line is, it’s still zero.

The basic functions for reading and writing strings are scanf and printf, which should be known to the class.

[Show some examples. Ask why you don’t need the &.]

There is a library of functions in the C standard library to manipulate strings.

[Give string library handout.]

[Demonstrate a few of them, including strlen(), strcpy(), strncmp().]

[Show a man page or two so they see how to get function usage.]

What is a command line argument?

It is anything typed after the name of the program. Each argument is separated by one or more spaces. Example:

```
int main(int argc, char *argv[])
{
int i;
for (i=0; i<argc; i++)
  printf("Argument %d is %s\n",i,argv[i]);
}
```

[Show additional examples as needed, and as time permits.]