Courtney Maynard, Assignment 10

1) For part one, I defined a function that takes ca as an argument and I created ca local to main.

The function:

```
void one (char ca[24] )
{
        printf(" addr of array param = %#x \n", &ca);
        printf(" addr of (ca[0]) = %#x \n", & (ca[0]));
        printf(" addr of (ca[1]) = %#x \n", &(ca[1]));
        printf(" ++ca = %#x \n\n", ++ca);
}
```

The creation of ca:

```
main()
{
         char ca[24];
```

2) For part two, I defined a function that takes pa as an argument and I created pa local to main.

The function:

```
void two (char *pa )
{
        printf(" addr of pointer param = %#x \n", &pa);
        printf(" addr of (pa[0]) = %#x \n", &(pa[0]));
        printf(" addr of (pa[1]) = %#x \n", &(pa[1]));
        printf(" ++pa = %#x \n\n", ++pa);
}
```

The creation of pa:

```
main()
{
       char ca[24];
       char *pa;
```

1-2) Main calls both one() and two(), with ca and pa the respective parameters for the function. I also included print statements so that I could better understand what outputs were being printed, due to many of the arguments being used as inputs multiple times. Including the print statements made it easier for me to understand and identify the similarities and differences in the addresses of the array vs pointers.

```
main()
{
        char ca[24];
        char *pa;

        printf("output of one() with ca as input to function:\n");
        one(ca);
        printf("output of two() with pa as input to function:\n");
        two(pa);
```

3) For part three, I created the global character array ga and had main call both one() and two() with ga as the parameter. ga is global and thus not local to the main function.

Creation of ga:

```
char ga[] = "abcdefghijklmnopqrstuvwxyz";
```

Main calling one() and two():

```
printf("output of one()  with ga as input to function:\n");
one(ga);
printf("output of two()  with ga as input to function: \n");
two(ga);
```

Comparing the outputs shows that the address of ga is the same for when it is represented as an array and when it is represented with a pointer to it. The components, such as pa[0] and pa[1], have the same addresses in memory whether ga was transformed into a pointer or whether it stayed as an array.

```
output of one()  with ga as input to function:
 addr of array param = 0x9443c9c8
 addr of (ca[0]) = 0x63c1b010
 addr of (ca[1]) = 0x63c1b011
 ++ca = 0x63c1b011

output of two()  with ga as input to function:
 addr of pointer param = 0x9443c9c8
 addr of (pa[0]) = 0x63c1b010
 addr of (pa[1]) = 0x63c1b011
 ++pa = 0x63c1b011
```

4) For part four, I printed the specified values.

Code for the printing of the values:

```
printf("addr of global array = %#x \n", &ga);
printf("addr (ga[0]) = %#x \n", &(ga[0]));
printf( "addr (ga[1]) = %#x \n\n", &(ga[1]));
```

Output:

```
addr of global array = 0x63c1b010
addr (ga[0]) = 0x63c1b010
addr (ga[1]) = 0x63c1b011
```

5) I expected that the address of the array parameter and the address of the pointer parameter when the respective functions, one and two, are executing, would be the same. I was correct in this prediction. My hypothesis is that this specific place in memory must be where the parameter passed to a function is stored. It is recycled afterwards and can be the memory address for a second function's parameter because the variables are created locally to the main function. On the other hand, the address of the global array itself is in a different location because it is declared globally and so it occupies some defined space in memory. I expected the address of the global array and the address of the first element in the global array to be the same, as well as the

address of the character array and the address of the first element of the character array to be the same. I was correct about the global array addresses, however, I was incorrect about the character array. The mis-match of addresses makes sense because of the call-by property of the arrays. The output and all the addresses of the different components of this assignment can be seen below:

```
cbmaynard24@th121-13:~/lbp/life/ASSIGNMENT_10$ ./a.out
output of one() with ca as input to function:
 addr of array param = 0x9443c9c8
 addr of (ca[0]) = 0x9443c9f0
 addr of (ca[1]) = 0x9443c9f1
 ++ca = 0x9443c9f1

output of two() with pa as input to function:
 addr of pointer param = 0x9443c9c8
 addr of (pa[0]) = 0x63c18370
 addr of (pa[1]) = 0x63c18371
 ++pa = 0x63c18371

output of one()  with ga as input to function:
 addr of array param = 0x9443c9c8
 addr of (ca[0]) = 0x63c1b010
 addr of (ca[1]) = 0x63c1b011
 ++ca = 0x63c1b011

output of two()  with ga as input to function:
 addr of pointer param = 0x9443c9c8
 addr of (pa[0]) = 0x63c1b010
 addr of (pa[1]) = 0x63c1b011
 ++pa = 0x63c1b011

addr of global array = 0x63c1b010
addr (ga[0]) = 0x63c1b010
addr (ga[1]) = 0x63c1b011
```