Justin Keeling
Homework 3
Chapter 9 Exercises

#2 In most Fortran IV implementations, all parameters were passed by reference, using access path transmission only. State both the advantages and disadvantages of this design choice.

Advantages: The load in which the compiler is tasked with, in terms of memory allocation, is light. The compiler does not have the responsibility of allocating more memory locations for the parameters, as it only passes in the specific memory location of the parameter. This leads to a faster access time of parameters as well. Thus writeability is a pro.
Disadvantages: Recursion is "impossible" or at least, difficult. Because recursion requires a stack based structure in order to use variable correctly, passing by reference will ultimately ruin this "stack based" structure that is needed (since the actual memory location is passed in as a parameter). Another disadvantage to this is readability. Because variable values can be changed outside its scope, keeping track of its value can be difficult.

#4 Suppose you want to write a method that prints a heading on a new output page, along with a page number that is 1 in the first activation and that increases by 1 with each subsequent activation. Can this be done without parameters and without reference to nonlocal variables in Java? Can it be done in C#?

Due to Java's and C# object oriented programming structure as well as static variables, it is possible to do this. Static variables allow different objects to share the same variable. Thus without parameters and references , it is possible because all objects share the same variable.

#5 Consider the following program written in C syntax:

```
void swap(int a, int b) {
  int temp;
  temp = a;
  a = b;
  b = temp;
}
void main() {
  int value = 2, list[5] = {1, 3, 5, 7, 9};
  swap(value, list[0]);
  swap(list[0], list[1]);
  swap(value, list[value]);
}
```

For each of the following parameter-passing methods, what are all of the values of the variables value and list after each of the three calls to swap?

1.Passed by value

- value = 2 , list[] = {1,3,5,7,9} for all three swap functions.

2.Passed by reference

- first swap : value = 1, list[] = {2,3,5,7,9}

- second swap : value = 1, list [] = {3,2,5,7,9}

- third swap : value = 2, list[] = {3,1,5,7,9}

3.Passed by value-result

- first swap : value = 1, list[] = {2,3,5,7,9}

- second swap : value = 1, list[] = {3,2,5,7,9}

- third swap : value = 2, list[] = {3,1,5,7,9}

#7 Consider the following program written in C syntax:

```
void fun (int first, int second) {
  first += first;
  second += second;
}
void main() {
  int list[2] = {1, 3};
  fun(list[0], list[1]);
}
```

For each of the following parameter-passing methods, what are the values of the list array after execution?

1. Passed by value

- list[] = {1,3}

2. Passed by reference

- list[] = {2, 6}

3. Passed by value-result

- list[] = {2,6}