Taylor Roberts
CS354
Ta2-2

Q: 3.2

Recursion was not supported in Fortran 77, so it was impossible to have multiple invocations of a subroutine. Variables were statically allocated, which caused issues running multiple sub-routines. Algol and later languages got rid of only static variables to fix this constraint.

Q: 3.4

1)

```
int main() {

int i=10;

{

int i=20;

{

int z=30;    //int z can only be called within this scope

printf(“” + i + “” + z);

}
}
}
```

2)

```
foo(){
      fee(){
      int b=1; //int b can only be called within this scope
      }
}
void fah(){
      int ha=0;
      //int b cannot be called her, for example
}
```

3)

```
int main() {
int b=1;
int a=0;
{
int b=2;
{
a=3; //even though int a can be called within in sub-scope, the current
value of the first a does not change until the interpreter scans the
second call to a;
}
}
}
```

3.5)

1, 1
3, 1
3, 1
3, 3
3, 3
3, 3
3, 3

3.7)

a) A new list is being made within the reverse method. Brad needs to deallocate the old list to make more memory space for the new list.
b) He has caused a dangling reference. The objects were deallocated too soon.

3.14)

Static: 1 1 2 2
Dynamic: 1 1 2 1

The second method acts upon different variables depending on the static or dynamic scoping. For static scope, the method manipulates the global variable x, however, dynamic scoping acts upon the variable within the scope only.

3.18)

Shallow: 1 0 2 0 3 0 4 0
Deep: 1 0 5 2 0 0 4 4

If the program has shallow binding, it can only access foo's x value. When utilizing deep binding, it can access both the global variable x and local variable x.