Conceptual Questions

1.)

* read
  + r or rb
* write
  + w or wb
* read and write
  + r+ or rb+ or r+b
* append
  + a+ or ab+ or a+b

2.)

* dynamic memory uses the heap
  + allocated at startup
  + reclaimed when the process exits
* the stack is used as scratch space for thread execution
  + always LIFO, to make sure that processes are executed in the correct order
  + allows for simple tracking of which block to use in the stack, as it just involves moving a pointer
  + stacks are attached to a thread, so when threads exit their stack is reclaimed in memory

3.)

* Pointers are variables that hold the address of another variable as their value
* Pointers in C are defined as:
  + type \*name;
  + type is the type of the variable the pointer is pointing too
  + name is the name of the pointer
* to set the value a pointer points to, use name = &var
  + name is the pointer, var is the variable address to have name point to
  + to get the address, you can now access the value of name

4.)

* malloc allocates memory of the given size and gives a
* to use malloc, you can initialize arrays of variables, such as characters, by initializing with:
  + var = (char \*) malloc(15)
  + this would allocate 15 character bytes at the returned address
* you can free the memory at a point by calling free(pointer), where pointer is the pointer variable pointing to the memory location you want to clear

5.)

* malloc takes 1 argument (number of bytes), calloc takes 2 arguments (number of blocks, and the size of the blocks)
* malloc returns a pointer to the given number of bytes of uninitialized storage, or NULL if it cannot allocate the given number of bytes in storage
* calloc returns a pointer to enough free space for an array of n objects of the specified size, or NULL