CSCI S-38 Summer 2018
Problem Set 4

1.
Part I
Redo the Fibonacci sequence problem, but this time use pointers and an array. The array size should be 30. Store the first 30 elements in the sequence in the array. Display the array. As a reminder, the sequence is: 0 1 1 2 3 5 8 … (each item is the sum of the previous 2 items).

Part II
Ask the user in advance how many numbers they want in the series. Dynamically allocate memory to store the numbers.
Create a function called makeFibSeries that takes a pointer to the beginning of the allocated block and how many numbers to compute.
In main(), ask the user for how many numbers, allocate the memory, call makeFibSeries() and then display the numbers.

2. Create a function called allocName. This function should take as input a first name and a last name. It should then dynamically allocate a memory block to store the full name and return a pointer to the beginning of the memory block.
In main(), call allocName, display the full name and then deallocate the memory block.

3. Create a structure called money that contains an int for dollars and cents.
Create a function called addMoney() that takes three parameters:
a struct of type money and an amount of dollars and cents to add (dollars and cents are separate int's)
create a function called showMoney() that displays a money structure as $d.cc
In main() create some money, add $3.75 to it using addMoney, display the resulting amount using showMoney()

4. Write a function called subString that returns a char *. It will take three input parameters:
a char * and two int's: index and howmany.
The function should return a substring that is formed from the *index* position of the input string for *howmany* characters or until the termination character ('\0') is encountered.
howmany should default to creating a substring that continues to the termination character.

Example:

```
char s1[80] = "Hello There");
subString(s1, 3, 4);  // returns "lo T"
subString(s1, 3, 99);  // returns "lo There"
subString(s1, 3);     // returns "lo There"
subString(s1);       // should not compile
```

Return a pointer to the beginning of the substring and insert a termination character into the string to create an endpoint.

Extra credit (required for grad students) Create an alternate form of the function that allocates sufficient memory for the substring and copies the substring to the new allocated space (you should use strcpy). The return value should be a pointer to the new memory block.