1. Write a *Stack* class.
    a. private members: *array , head*
    b. public member functions:
        i. *void push(),*
        ii. *void pop(),*
        iii. *int peek(),*
        iv. *int  getSize(),*
        v. *bool isEmpty()*

2. Correct the previous code.
    a. Write *init* function.
3. Write a function that prints out all the elements of the stack.
    *a. void dumpStack(Stack s){}*
4. Write a function that takes as argument a stack and makes a copy of the given stack. The function will return the copy.
    *a. Stack copyStack(Stack s){}*
5. Write constructor function. (Empty constructor)
6. Write destructor function.
7. Update the definition of *Stack*.
    a. Instead of a fixed length array use a pointer.
    b. Update init function.
    c. Write constructor function. (use *new*)
    d. Write destructor function. (use *delete*)
    e. Perform step practice problem 3 and 4 for the updated stack. Is it working ?
    f. Instead of using the *copyStack()* function assign one stack to the using '=' operator. Then call *dumpStack()* with the new stack.

////////////////----------XXXXXX----------////////////////

8. Write a *String* class.
    a. private member:
        i. Array of characters.
        ii. Length
    b. Write constructor. Takes a string as parameter.
        *i. String(char *s){}*
    c. public member function:
        *i. void concat(char *s)*
    d. Overload *concat*
        *i. void concat(int a);*
        *ii. void concat(double d);*
        *iii. void concat(char c);*
        *iv. void concat(int a, char c);*
    e. Overload
        i. Empty Constructor.
        ii. Takes a string as input.
        iii. Takes desired *length* of the string and creates a string will *length* number of spaces .

9. Update *String* class.
    a. private member:
        i. Pointer to charater.
        ii. Length
    b. Update all the overloaded constructors.
    c. Use concat.
    d. Write Copy Constructor.
    e. Use concat.