MATH 626 Project 1 - Dynamic Array Stack

Using ArrayStackFixed.c as a model, implement the stack abstract data type with an underlying dynamically allocated array. That is, use the `malloc` and `realloc` functions in `stdlib.h`. Ensure your project meets the following specifications

- In comments at the top of the file, briefly describe how and when the dynamically allocated array is expanded/shrunk. Be sure your code matches your documentation. You can come up with your own scheme for resizing, for example, you could resize the array in increments of 100, or double it/half it as required.

- Instead of holding characters, the stack should hold `double` data vales.

- Following ArrayStackFixed.c, you should include functions to both initialize, and destroy the stack.

- Ensure there is a function with prototype `void print(ArrStack *);` that prints the contents of the stack to the console.

- Be sure the push, pop, and peek functions have the following definitions/prototypes

```
void push(ArrStack *, double);
double pop(ArrStack *);
double peek(ArrStack *);
```

- You will upload a single file to Canvas please ensure it is named "LastName - Project1.c"

- I plan on testing your stack with code similar to that below

```
//testing
ArrStack *myStack = newStack();

push(myStack, 3.4);
push(myStack, 6.7);
push(myStack, 6.7);
print(myStack);
printf("%lf\n", pop(myStack));
printf("%lf\n", pop(myStack));
printf("%lf\n", pop(myStack));
pop(myStack);
pop(myStack);
print(myStack);
push(myStack, 4.7);
```

```
push(myStack, 2.45);
push(myStack, 4.654);
peek(myStack);
printf("%lf\n", peek(myStack));
printf("%lf\n", peek(myStack));
printf("%lf\n", pop(myStack));
printf("%lf\n", peek(myStack));
printf("%lf\n", pop(myStack));
printf("%lf\n", peek(myStack));
printf("%lf\n", pop(myStack));
printf("%lf\n", peek(myStack));;
destroyStack(myStack);
```