Tester: Osäker på exakt vad ni menar men har implementerat inmatning från användaren på alla uppgifter samt vissa instruktioner om användaren får val på vad de vill göra. Kan visa detta på redovisningstillfället eller om ni ber mig visa. Gjort massa tester själv via min implementation och testkört att programmet fungerar som önskat.

Val av algoritm p 1.2.

Valde stack pga den är extremt smidig då för att skriva ut en char array baklänges. Det ända man behövde göra är att pusha in elementen från 0 -> n i stacken så när man .pop:ar ut dom så poppas dom översta elementet först ned till alla element är popade. Alltså n->0 vilket är det vi vill. Det är bara att se till att man skriver ut det ropade items så kommer en inmatad stäng att skrivas ut baklänges. Kan förklara mer utförligt vid redovisningstillfället om det behövs.

Komplexiteten:

Se headern I Lab1.7

Men

Räknade allt I worst case.

Enligt min algoritm kommer tidskomplexiteten vara som värst O(n) för att i värsta fall måste programmet gå igenom hela char arraken som består av n element. Bokstavligen en ända for loop som går från 0 -> array.lenght.

Minneskomplexiteten kommer också vara O(n) då i värsta fall måste alla element att pushas in i stacken. En snygg lösning till det problemet vore vara att göra en probe som kollar hur många element som krävs för att vara balanserad och hur många som finns kvar etc. Har inte hunnit implementera detta så min har O(n).