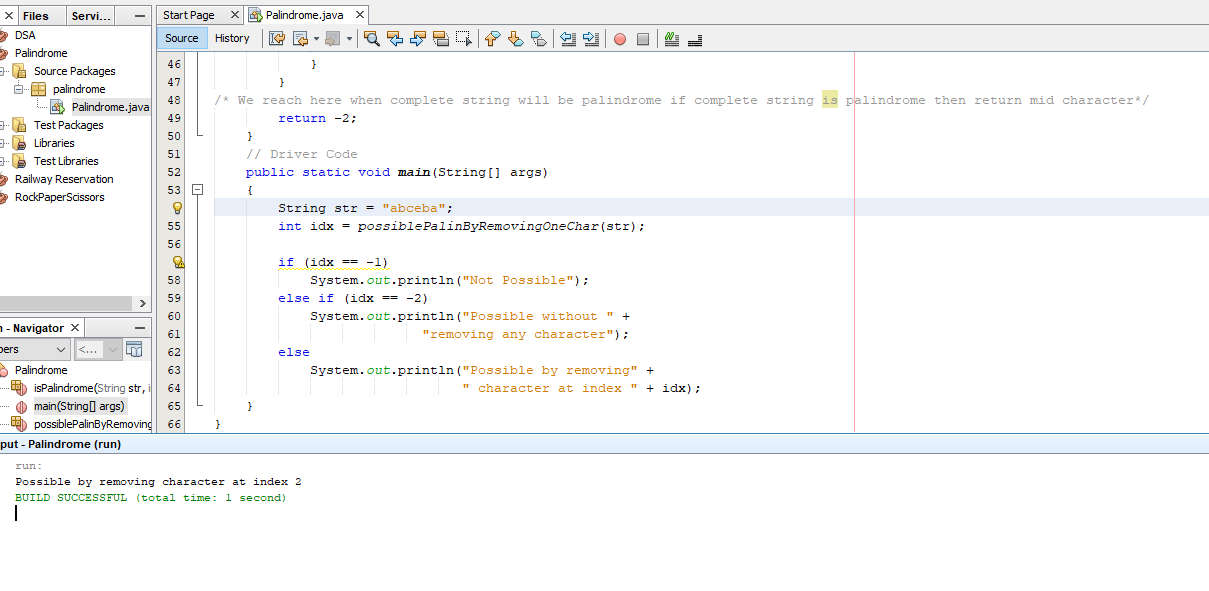
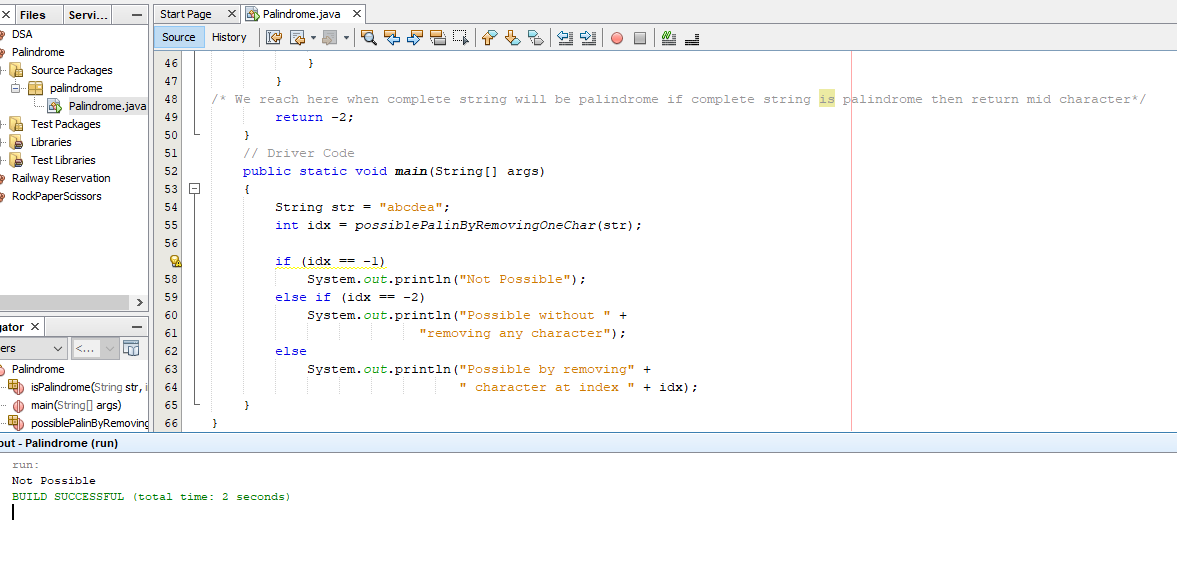
GROUP MEMBERS

1. 134126 Adrianna Bitutu
2. 127707 Clarice Muthoni
3. 130834 Alvin Maina
4. 124534 Charity Makhanu

**Question One: Develop and implement an algorithm which checks if a string given by a user can be converted into a palindrome by removing a defined number of characters.**

1. Utility to check if string is palindrome or not
2. returns -1 if it is not possible to make string a palindrome.
3. returns -2 if string is already a palindrome. Otherwise it returns index of character whose removal can make the whole string palindrome.
4. loop untill low and high cross each other
5. If both characters are equal then move both pointer towards end
6. If removing str[low] makes the whole string palindrome. We basically is palindrome or not.check if substring str[low+1..high]
7. If removing str[high] makes the whole string palindrome. We basically check if substring str[low+1..high] is palindrome or not.
8. if complete string is palindrome then return mid character

return -2;

**Question Two: Develop an algorithm that aids in the creation of non-empty palindromes**

**i.e. an algorithm that takes a user input string and produces a pre-set number of palindromes e.g. Given a string "annabelle" your algorithm should form 3 (or more)**

**palindromes such as ‘anna’, ‘elle’ & ‘b’.**

1. Initial Values : i = 0, j = n-1;
2. Given string 'str'
3. CountPS(i, j)
4. If length of string is 2 then we check both character are same or not
5. If (j == i+1) return str[i] == str[j] this condition shows that in recursion if i crosses j then it will be a invalid substring or if i==j that means only one character is remaining and we require substring of length 2 in both the conditions we need to return 0
6. Else if(i == j || i > j) return 0;
7. Else If str[i..j] is PALINDROME
8. increment count by 1 and check for rest palindromic substring (i, j-1), (i+1, j) remove common palindrome substring (i+1, j-1)
9. return countPS(i+1, j) + countPS(i, j-1) + 1 - countPS(i+1, j-1)
10. Else if NOT PALINDROME We check for rest palindromic substrings (i, j-1) and (i+1, j) remove common palindrome substring (i+1 , j-1)
11. return countPS(i+1, j) + countPS(i, j-1) - countPS(i+1 , j-1);

