Gru has a string *S* of length *N*, consisting of only characters *a* and *b* for banana and *P* points to spend.

Now Gru wants to replace and/or re-arrange characters of this given string to get the lexicographically smallest string possible. For this, he can perform the following two operations any number of times.

1. Swap any two characters in the string. This operation costs 11*point*. (any two, need not be adjacent)
2. Replace a character in the string with any other lower case english letter. This operation costs 2 *points*.

Help Gru in obtaining the lexicographically smallest string possible, by using at most *P* points.

Input:

- First line will contain *T*, number of testcases. Then the testcases follow.
- Each testcase contains two lines of input, first-line containing two integers *N* , *P*.
- The second line contains a string *S* consisting of *N* characters.

Output: For each testcase, output in a single containing the lexicographically smallest string obtained.

Sample Input

1

3 3

bba

Sample Output

aab

```
import java.util.*;
import java.lang.*;
import java.io.*;

class Program {
	public static void main(String[] args) throws java.lang.Exception {
```

```
		BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
		int t1 = Integer.parseInt(br.readLine());
		for (int t = 1; t <= t1; t++) {
			StringTokenizer tk = new StringTokenizer(br.readLine());
			int n = Integer.parseInt(tk.nextToken());
			int p = Integer.parseInt(tk.nextToken());
			String s = br.readLine();
			char c[] = s.toCharArray();
			int b = 0;
			for (int i = 0; i < c.length; i++) {
				if (s.charAt(i) == 'b')
					b++;
			}
			int a1 = 0;
			for (int i = c.length - 1; i >= n - b; i--) {
				if (c[i] == 'a')
					a1++;
			}
			if (p <= a1) {
				int p1 = p;
				for (int i = 0; i < c.length && p1 > 0; i++) {
					if (c[i] == 'a')
						continue;
					else {
						c[i] = 'a';
						p1--;
					}
				}
				p1 = p;
				for (int i = n - 1; i >= 0 && p1 > 0; i--) {
					if (c[i] == 'b')
						continue;
					else {
						c[i] = 'b';
						p1--;
					}
				}
				System.out.println(String.valueOf(c));
			} else {
				char c2[] = s.toCharArray();
				Arrays.sort(c2);
				int p1 = p - a1;
				for (int i = n - b; i < c.length && p1 > 0; i++) {
					if (c[i] == 'b') {
						if (p1 >= 2) {
							c2[i] = 'a';
							p1 -= 2;
						}
					} else {
						c2[i] = 'a';
						p1--;
					}
				}
				System.out.println(String.valueOf(c2));
```

```
                    }

                }
            }
        }
```