Problem BinSearch

Input file	stdin
Output file	stdout

```
bool binary_search(int n, int p[], int target){
    int left = 1, right = n;
    while(left < right){
        int mid = (left + right) / 2;
        if(p[mid] == target)
            return true;
        else if(p[mid] < target)
            left = mid + 1;
        else
            right = mid - 1;
    }
    if(p[left] == target) return true;
    else return false;
}
```

Είναι γνωστό ότι εάν ο p τυχαίνει να είναι ταξινομημένος, τότε αυτός ο κώδικας επιστρέφει true εάν και μόνο εάν το target εμφανίζεται εντός του p. Από την άλλη, αυτό μπορεί να μην ισχύει εάν ο p δεν έχει ταξινομηθεί.

Σας δίνεται ένας θετικός ακέραιος αριθμός n και μια ακολουθία $b_1, \ldots, b_n \in \{\texttt{true}, \texttt{false}\}$. Είναι εγγυημένο ότι $n = 2^k - 1$ για κάποιο θετικό ακέραιο k.

Πρέπει να δημιουργήσετε μια μετάθεση p του $\{1, \ldots, n\}$ που ικανοποιεί συγκεκριμένες συνθήκες. Έστω $S(p)$ ο αριθμός των δεικτών $i \in \{1, \ldots, n\}$ για τους οποίους η συνάρτηση binary_search(n, p, i) δεν επιστρέφει το b_i. Πρέπει να ορίσετε το p έτσι ώστε το $S(p)$ να είναι μικρό (όπως περιγράφεται στην ενότητα "Restrictions").

(Σημείωση: μια μετάθεση του $\{1, \ldots, n\}$ είναι μια ακολουθία n ακεραίων που περιέχει κάθε ακέραιο από το 1 έως το n ακριβώς μία φορά.)

Input data

Η είσοδος περιέχει πολλαπλά test case. Η πρώτη γραμμή εισαγωγής περιέχει το T, τον αριθμό των test cases. Ακολουθούν τα test cases.

Η πρώτη γραμμή ενός test case περιέχει τον ακέραιο αριθμό n. Η δεύτερη γραμμή ενός test case περιέχει μια συμβολοσειρά μήκους n που περιέχει μόνο τους χαρακτήρες "0" και "1". Αυτοί οι χαρακτήρες δεν χωρίζονται με κενά. Εάν ο i-οστός χαρακτήρας είναι "1", τότε $b_i = \texttt{true}$, και αν είναι "0", τότε $b_i = \texttt{false}$.

Output data

Τα δεδομένα εξόδου αποτελούνται από τις απαντήσεις για κάθε ένα από τα T test cases. Η απάντηση για ένα test case αποτελείται από τη μετάθεση p που δημιουργήθηκε για το test case.

Restrictions

- Έστω $\sum n$ είναι το άθροισμα όλων των τιμών n σε μία μόνο είσοδο.
- $1 \leq \sum n \leq 100\,000$.
- $1 \leq T \leq 7\,000$.
- $n = 2^k - 1$ για κάποιο $k \in \mathbb{N}$, $k > 0$.
- Εάν $S(p) \leq 1$ για όλα τα test cases σε ένα subtask, τότε βαθμολογείστε με 100% των πόντων για αυτό το subtask.
- Αλλιώς, εάν $0 \leq S(p) \leq \lceil \log_2 n \rceil$ (δηλαδή $1 \leq 2^{S(p)} \leq n+1$) για όλα τα test cases σε ένα subtask, τότε βαθμολογείστε με 50% των πόντων για αυτό το subtask.

#	Points	Restrictions
1	3	$b_i =$ `true`.
2	4	$b_i =$ `false`.
3	16	$1 \leq n \leq 7$.
4	25	$1 \leq n \leq 15$.
5	22	$n = 2^{16} - 1$ και κάθε b_i επιλέγεται ομοιόμορφα και ανεξάρτητα τυχαία από το σύνολο {`true`,
6	30	Χωρίς πρόσθετους περιορισμούς.

Examples

Input file	Output file
4 3 111 7 1111111 3 000 7 000000000	1 2 3 1 2 3 4 5 6 7 3 2 1 7 6 5 4 3 2 1
2 3 010 7 0010110	3 2 1 7 3 1 5 2 4 6

Explanations

Παράδειγμα 1 Στα δύο πρώτα test cases του πρώτου παραδείγματος έχουμε $S(p) = 0$.

Στο τρίτο test case, έχουμε $S(p) = 1$, καθώς η συνάρτηση `binary_search(n, p, 2)` επιστρέφει `true`, αν και $b_2 =$ `false`.

Στο τέταρτο test case, έχουμε $S(p) = 1$, καθώς η συνάρτηση `binary_search(n, p, 4)` επιστρέφει `true`, αν και $b_4 =$ `false`.

Παράδειγμα 2. Έχουμε $S(p) = 0$ και για τα δύο test cases.