C. Balance the Bits

time limit per test

1 second

memory limit per test

256 megabytes

input

standard input

output

standard output

A sequence of brackets is called balanced if one can turn it into a valid math expression by adding characters '+' and '1'. For example, sequences '(())()', '()', and '(()(()))' are balanced, while ')(', '(()', and '(()))(' are not.

You are given a binary string ss of length nn. Construct two balanced bracket sequences aa and bb of length nn such that for all 1≤i≤n1≤i≤n:

* if si=1si=1, then ai=biai=bi
* if si=0si=0, then ai≠biai≠bi

If it is impossible, you should report about it.

**Input**

The first line contains a single integer tt (1≤t≤1041≤t≤104) — the number of test cases.

The first line of each test case contains a single integer nn (2≤n≤2⋅1052≤n≤2⋅105, nn is even).

The next line contains a string ss of length nn, consisting of characters 0 and 1.

The sum of nn across all test cases does not exceed 2⋅1052⋅105.

**Output**

If such two balanced bracked sequences exist, output "YES" on the first line, otherwise output "NO". You can print each letter in any case (upper or lower).

If the answer is "YES", output the balanced bracket sequences aa and bb satisfying the conditions on the next two lines.

If there are multiple solutions, you may print any.

**Example**

**input**

**Copy**

3

6

101101

10

1001101101

4

1100

**output**

**Copy**

YES

()()()

((()))

YES

()()((()))

(())()()()

NO

**Note**

In the first test case, a=a="()()()" and b=b="((()))". The characters are equal in positions 11, 33, 44, and 66, which are the exact same positions where si=1si=1.

In the second test case, a=a="()()((()))" and b=b="(())()()()". The characters are equal in positions 11, 44, 55, 77, 88, 1010, which are the exact same positions where si=1si=1.

In the third test case, there is no solution.

给定一个字符串，仅由0和1组成，长度为n。你需要给出两个合法的括号序列，它们的长度为n。并且满足：如果字符串的第n位为1，则这两个括号序列的第n位相同。如果字符串的第n位为0，则这两个括号序列的第n位相反。如果不存在这样的括号序列，输出NO。

首先，合法的括号序列必须由'('开始，到')'结束。所以字符串开头和结尾必须都为1。然后，0的个数必须为偶数，否则两个序列的'('和')'的数量肯定不同，导致肯定有一个序列不合法。最后就是如何构造的问题。

一个比较简单的方法是，对字符串的前n/2个1，两个括号序列的对应位置都是'('。对字符串的后n/2个1，两个括号序列的对应位置都是')'。

对字符串的第奇数个0，两个括号序列的对应位置是'('和')'。对字符串的第偶数个0，两个括号序列的对应位置是')'和'('。

举几个例子：

10000001

(  ()()()  )

(  )()()(  )

从上面这个例子也可以很容易的理解为什么必须有偶数格0。

#include<bits/stdc++.h>

#define ll long long

using namespace std;

int main()

{

int t;

scanf("%d",&t);

while(t--)

{

int n;

scanf("%d",&n);

string a;

cin>>a;

int ct1=0;

int ct2=0;

for(int i=0;i<a.length();i++)

{

if(a[i]=='1') ct1++;

if(a[i]=='0') ct2++;

}

int flag=1;

string ans1="";

string ans2="";

if(ct1%2!=0) flag=0;

if(a[0]!='1'||a[a.length()-1]!='1') {flag=0;}

int c1=0;

int c2=0;

for(int i=0;i<a.length();i++)

{

if(a[i]=='1') c1++; else c2++;

if(a[i]=='1')

{

if(c1<=ct1/2) {ans1+='('; ans2+='('; continue;}

else {ans1+=')'; ans2+=')'; continue;}

}

else

{

if(c2%2==1) {ans1+='('; ans2+=')';}

else {ans1+=')'; ans2+='(';}

}

}

if(flag==0) printf("NO\n");

else {printf("YES\n");cout<<ans1<<"\n"<<ans2<<"\n";}

}

return 0;

}