Programming Problem: Check if All Digits are Even and Last Digit is Even

Problem Statement

Write a program that reads multiple strings representing numbers. For each number, determine if **all digits are even** and the **last digit is even**.

- If the last digit is even and **all digits** in the number are even, print ("YES").
- Otherwise, print ("NO").

Input Format

- The first line contains an integer (t) the number of test cases.
- The next (t) lines each contain a string (a) representing a number. The number can be up to 20 digits long.

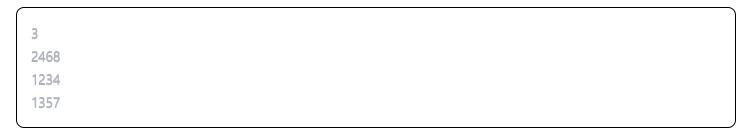
Output Format

For each test case, print a single line containing either ("YES") or ("NO") according to the rules above.

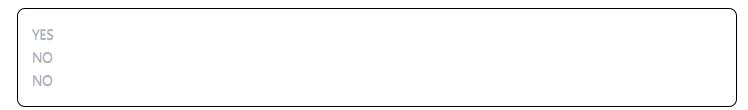
Constraints

- $1 \le t \le 1000$
- Each number (a) contains only digits (0)–(9).
- Length of $(a) \le 20$.

Sample Input



Sample Output



Explanation

- For 2468, the last digit is 8 (even), and all digits (2,4,6,8) are even \rightarrow ("YES").
- For (1234), the last digit is 4 (even), but not all digits are even (1 and 3 are odd) \rightarrow ("NO")
- For (1357), the last digit is 7 (odd) \rightarrow ("NO").

Implementation Notes

- You can read each number as a string to handle large numbers.
- Check the last character of the string to verify if it's even.
- Then check all characters to ensure they represent even digits.

Solution Approach

- 1. Read the number of test cases (t).
- 2. For each test case:
 - Read the number as a string.
 - Check if the last digit is even.
 - Check if all digits in the string are even.
 - Print "YES" if both conditions are met, otherwise print "NO"

Hint

Even digits are: 0, 2, 4, 6, 8 Odd digits are: 1, 3, 5, 7, 9

Sample Solution (C)



```
#include <stdio.h>
#include <string.h>
void solve() {
  char a[21]; // Size 21 to handle up to 20 digits + null terminator
  fgets(a, sizeof(a), stdin); // Safer than gets()
  // Remove newline character if present
  int len = strlen(a);
  if (a[len-1] == '\n') {
    a[len-1] = '\0';
    len--;
  // Check if last digit is even
  if ((a[len - 1] - '0') \% 2 == 0) {
    // Check if all digits are even
     for (int i = 0; i < len; i++) {
       if ((a[i] - '0') % 2!= 0) {
          printf("NO");
          return;
     printf("YES");
  } else {
     printf("NO");
int main() {
  int t;
  scanf("%d", &t);
  getchar(); // Clear the newline after scanf
  while (t--) {
     solve();
     printf("\n");
  return 0;
```

Key Points in the Solution

- 1. **Character to Digit Conversion**: Use (a[i] '0') to convert character to actual digit value
- 2. **Safe Input**: Use (fgets()) instead of deprecated (gets())
- 3. **Buffer Size**: Array size should be 21 to handle 20 digits + null terminator
- 4. **Newline Handling**: Remove newline character that (fgets()) might include
- 5. **Algorithm**: Check last digit first, then iterate through all digits

Common Mistakes to Avoid

- **ASCII vs Digit Value**: Don't use (a[i] % 2) directly characters have ASCII values, not digit values
- **Buffer Overflow**: Ensure array size is adequate (length + 1 for null terminator)
- **Input Handling**: Clear newline after (scanf()) with (getchar())