Q.1. Write a program to check whether a string is pangram or not

A sentence is a pangram if it contains every letter of the alphabet i.e. a-z

Test cases:

1. Input: quick brown fox jumps over the lazy dog
   Output: pangram
2. Input: 123456$#Hello
   Output: not a pangram

Refute Test cases:

1. Input: Jack quickly enjoyed the vibrant, mixed flavors of a cozy, warm pizza with extra cheese on top
   Output: not a pangram
2. Input: Zoe quickly juggled six very large boxes full of exotic French pastries.
   Output: not a pangram

Here, both the sentences in Refute Test cases have all alphabets but shows not a pangram because it exceeds 50 letters

In C, when we take input for string str[50] , it records letters till 50 characters and ignores the rest. Hence, only the 50 letters are stored and not the whole sentence since it exceeds the limit of how much this character array can store.

Program:

```
#include <stdio.h>
#include<string.h>
#include<ctype.h>
int pangram(char*);
int pangram(char* arr) {
    int i = 0, c = arr[0];
    int letter[26] = {0};
    //traversal through string %[^\n]%c
    for(i=0;arr[i];i++) {
        c=tolower(arr[i]);
        if(isalpha(arr[i]))
        letter[c-'a']=1;
    }
    //check for 1's
    for(i=0;i<26;i++) {
        if(!letter[i]) return 0;
    }
    return 1;
}
int main()
{
    char str[50];
    scanf("%[^\n]s", str); //alternative to %*[^\n]%*c
    (pangram(str)) ? printf("pangram") : printf("not a pangram");

            return 0;
}
```

Output:

Test Cases :

Input 1 :

```
Enter sentence: quick brown fox jumps over the lazy dog
pangram
```

Input 2 :

```
Enter sentence: 123456$#Hello
not a pangram
```

Refute Test Cases :

Input 1 :

```
Enter sentence: Jack quickly enjoyed the vibrant, mixed flavors of a cozy, warm pizza with extra cheese on top
not a pangram
```

Input 2 :

```
Enter sentence: Zoe quickly juggled six very large boxes full of exotic French pastries.
not a pangram
```

Q.2. Write a program to display number of valid pairs of X and Y where X and Y are number of odd and even pairs in the range
1<=X<=U
1<=Y<=V
(where U and V are inputs by user)

Test cases:

1. **Input:**
   25 50
   Output:
   625
2. **Input:**
   17 5
   Output:
   43

Refute Test cases:

1. Input:
   hello
   Output:
   -727241856
2. Input:
   25 h8
   Output:
   52511000

The input for refute cases is string which is invalid since its not an integer value.

Hence, it throws a ValueError

Program:

```
#include<stdio.h>
int main() {
    //for U
    int U,V;
    scanf("%d%d",&U,&V);
    int evenX = U/2;
    int oddX = U - evenX;
    //for V
    int evenY = V/2;
    int oddY = V - evenY;
    int valid_pairs = evenX * evenY + oddX * oddY;
    printf("%d",valid_pairs);
    return 0;
}
```

Output:

Test Cases :

Input 1 :

```
25 50
625
```

Input 2 :

```
17 5
43
```

Refute Test Cases :

Input 1 :

```
25 h8
52511000
```

Input 2 :

```
hello
-727241856
```