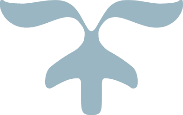


DAA WEEK – 6 SKILL – 6



# [Pangrams](https://www.hackerrank.com/contests/daa-skill-05-strings-and-pattern-matching-part-2/challenges/pangrams)

#include <stdio.h>

#include <string.h>

#include <ctype.h>

#define ALPHABET\_SIZE 26

const char\* pangrams(const char\* s) {

int alphabet[ALPHABET\_SIZE] = {0};

int index;

for (int i = 0; s[i]; i++) {

if (isalpha(s[i])) {

index = tolower(s[i]) - 'a';

alphabet[index] = 1;

}

}

for (int i = 0; i < ALPHABET\_SIZE; i++) {

if (alphabet[i] == 0) {

return "not pangram";

}

}

return "pangram";

}

int main() {

char s[1000];

fgets(s, sizeof(s), stdin);

const char\* result = pangrams(s);

printf("%s\n", result);

return 0;

}

**Pangrams Test Cases**

**A screenshot of a computer

Description automatically generated**

# [Separate the Numbers](https://www.hackerrank.com/contests/daa-skill-05-strings-and-pattern-matching-part-2/challenges/separate-the-numbers)

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

void separateNumbers(char\* s) {

int f = 0;

long w = 0;

int len = strlen(s);

for (int x = 1; x <= len / 2; x++) {

char a[20];

strncpy(a, s, x);

a[x] = '\0';

long q = strtol(a, NULL, 10);

w = q;

char b[100] = "";

while (strlen(b) < len) {

char temp[20];

sprintf(temp, "%ld", q++);

strcat(b, temp);

}

if (strcmp(b, s) == 0) {

printf("YES %ld\n", w);

f++;

break;

}

}

if (f == 0) {

printf("NO\n");

}

}

int main() {

int q;

scanf("%d", &q);

for (int a0 = 0; a0 < q; a0++) {

char s[100];

scanf("%s", s);

separateNumbers(s);

}

return 0;

}

**Separate the Numbers Test Cases**

**A screenshot of a computer

Description automatically generated**

# [Funny String](https://www.hackerrank.com/contests/daa-skill-05-strings-and-pattern-matching-part-2/challenges/funny-string) Python 3

import os

def funnyString(s):

s\_rev = s[::-1]

l1 = [abs(ord(s[i]) - ord(s[i+1])) for i in range(len(s)-1)]

l2 = [abs(ord(s\_rev[i]) - ord(s\_rev[i+1])) for i in range(len(s\_rev)-1)]

if l1 == l2:

return 'Funny'

return 'Not Funny'

if \_\_name\_\_ == '\_\_main\_\_':

fptr = open(os.environ['OUTPUT\_PATH'], 'w')

q = int(input().strip())

for q\_itr in range(q):

s = input()

result = funnyString(s)

fptr.write(result + '\n')

fptr.close()

**Funny String Test Cases**

**A screenshot of a computer

Description automatically generated**

# [Gemstones](https://www.hackerrank.com/contests/daa-skill-05-strings-and-pattern-matching-part-2/challenges/gem-stones)

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int gemstones(char arr[][101], int n) {

int mineral\_count[26] = {0};

int common\_count = 0;

for (int i = 0; i < n; i++) {

int found[26] = {0};

for (int j = 0; arr[i][j] != '\0'; j++) {

if (!found[arr[i][j] - 'a']) {

found[arr[i][j] - 'a'] = 1;

mineral\_count[arr[i][j] - 'a']++;

}

}

}

for (int i = 0; i < 26; i++) {

if (mineral\_count[i] == n) {

common\_count++;

}

}

return common\_count;

}

int main() {

int n;

scanf("%d", &n);

char arr[n][101];

for (int i = 0; i < n; i++) {

scanf("%s", arr[i]);

}

int result = gemstones(arr, n);

printf("%d\n", result);

return 0;

}

**Gemstones Test Cases**

**A screenshot of a computer

Description automatically generated**

# [Alternating Characters](https://www.hackerrank.com/contests/daa-skill-05-strings-and-pattern-matching-part-2/challenges/alternating-characters)

#include <stdio.h>

#include <string.h>

int alternatingCharacters(char\* s) {

int deletions = 0;

int len = strlen(s);

for (int i = 1; i < len; i++) {

if (s[i] == s[i - 1]) {

deletions++;

}

}

return deletions;

}

int main() {

int q;

scanf("%d", &q);

for (int i = 0; i < q; i++) {

char s[100000];

scanf("%s", s);

int result = alternatingCharacters(s);

printf("%d\n", result);

}

return 0;

}

**Alternating Characters Test Cases**

**A screenshot of a computer

Description automatically generated**

**SKILL WEEK – 6**

[**https://www.hackerrank.com/contests/daa-skill-05-strings-and-pattern-matching-part-2/challenges/alternating-characters**](https://www.hackerrank.com/contests/daa-skill-05-strings-and-pattern-matching-part-2/challenges/alternating-characters)