ASSIGNMENT 1

QUESTION 1

Given a string **s** and a number **x**, print the shortest substrings which start and end with the same character and have lengths greater than or equal to **x**. If multiple substrings exist with the same shortest length, print them all.

Eg:

s: abccdbacca

x: 3	Answer: acca
x: 4	Answer: acca
x: 5	Answer: bccdb cdbac
x: 6	Answer: abccdba
x: 7	Answer: abccdba
x: 8	Answer: not-found

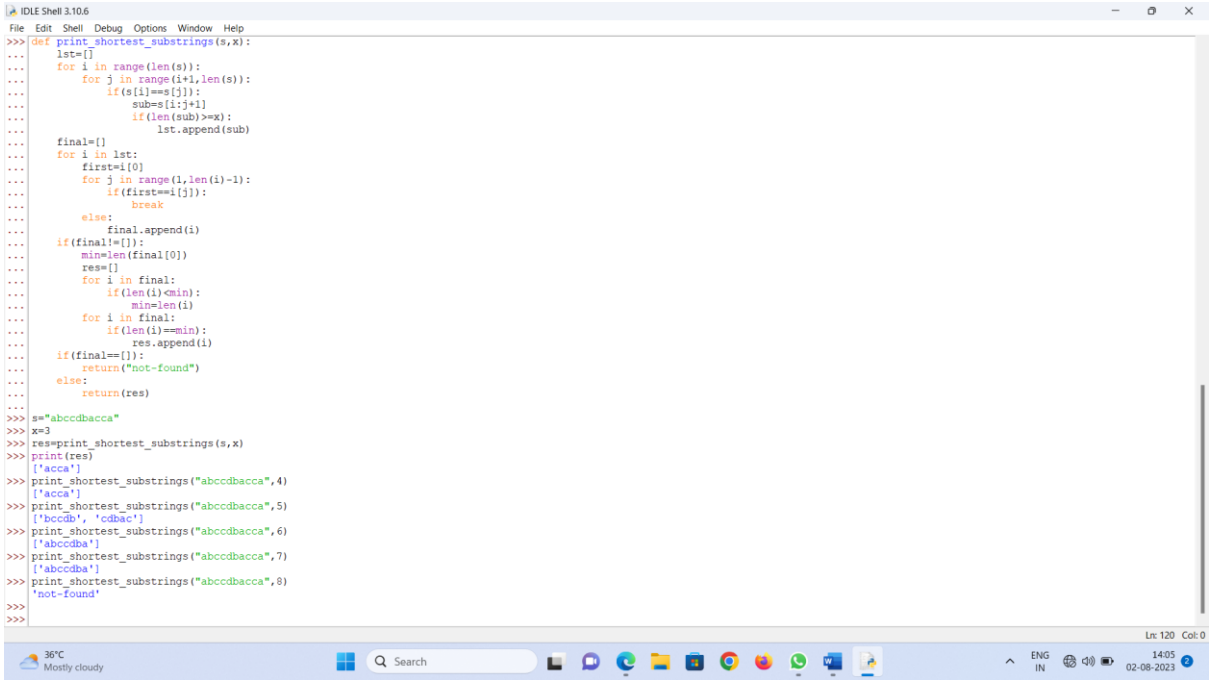
```
# Example usage
s = "abccdbacca"
x = 3
print("x =", x)
print_shortest_substrings(s, x)

x = 4
print("\nx =", x)
print_shortest_substrings(s, x)

x = 5
print("\nx =", x)
print_shortest_substrings(s, x)

x = 6
print("\nx =", x)
print_shortest_substrings(s, x)

x = 7
print("\nx =", x)
print_shortest_substrings(s, x)

x = 8
print("\nx =", x)
print_shortest_substrings(s, x)
```

PROGRAM CODE (Python):

```
def print_shortest_substrings(s,x):
    lst=[]
    for i in range(len(s)):
        for j in range(i+1,len(s)):
            if(s[i]==s[j]):
                sub=s[i:j+1]
                if(len(sub)>=x):
                    lst.append(sub)
    final=[]
    for i in lst:
        first=i[0]
        for j in range(1,len(i)-1):
            if(first==i[j]):
                break
        else:
            final.append(i)
    if(final!=[]):
        min=len(final[0])
        res=[]
        for i in final:
            if(len(i)<min):
                min=len(i)
        for i in final:
            if(len(i)==min):
                res.append(i)
    if(final==[]):
        return("not-found")
    else:
        return(res)
```

```
s="abccdbacca"
x=7
res=print_shortest_substrings(s,x)
print(res)
```

OUTPUT:



QUESTION 2

Given a string **s**, find the ASCII value of each character iteratively. If the ASCII value is even, increment the next character by (**ASCII_value % 7**). If the ascii value is odd, decrement the previous character by (**ASCII_value % 5**). Output the newly formed string.
Note:

- If a character has already been changed once, do not change that character again.
- If the new number is an invalid ASCII value, **replace it with 83**.

Eg:

s: sHQen}
ASCII: 115-72-81-101-110-125

First pass (115): *No previous character.*
115-72-81-101-110-125
Second pass (72): *Increment the next character by (**72%7**)*
115-72-**83**-101-110-125
Third pass(83): *Decrement previous character by (**83%5**)*
115-**69**-**83**-101-110-125
Fourth pass (101): *Previous character already changed once.*
115-**69**-**83**-101-110-125
Fifth pass(110): *Invalid ASCII value.*
115-**69**-**83**-101-110-**83**
Sixth pass(83):
115-**69**-**83**-101-**107**-**83**

Final Answer:
115-**69**-**83**-101-**107**-**83** => sESekS

PROGRAM CODE (Python):

```
def fun(s):
    old=[]
    lst=[]
    for i in s:
        old.append(ord(i))
        lst.append(ord(i))
    for i in range(1,len(lst)):
        if(lst[i]%2==0):
            if(lst[i+1]==old[i+1]):
                x=lst[i+1]+(lst[i]%7)
                if(x>=32 and x<=126):
```

```
                lst[i+1]=x
            else:
                lst[i+1]=83
        else:
            if(lst[i-1]==old[i-1]):
                x=lst[i-1]-(lst[i]%5)
                if(x>=32 and x<=126):
                    lst[i-1]=x
                else:
                    lst[i-1]=83
    string=""
    for i in lst:
        string=string+chr(i)
    print(string)

s="sHQen}"
fun(s)
```