BÁO CÁO ĐÁNH GIÁ BÀI TẬP VỀ NHÀ NHÓM 09

1. Nhóm 1:

```python
s = input()
rev = s[::-1]
res = ''
for i in range(len(s)):
    for j in range(i + 1, len(s) + 1):
        res = s[i:j] if (len(s[i:j]) > len(res)) and s[i:j] == rev[len(s)-j:len(s)-i] else res
print(len(res), res)
```

Nhóm làm đúng phương pháp thiết kế thuật toán Brute Force, tuy nhiên chưa kiểm tra trường hợp chuỗi rỗng nên chưa đạt điểm tối đa ở lần nộp này.

```python
if __name__ == "__main__":
    try:
        s = input()
        longestPalSubstr(s)
    except EOFError:
        print(0)
```

Sau đó, nhóm có sửa lại và đạt điểm tối đa.

2. Nhóm 2:

```python
def is_palindrome(str):
    return str == str[::-1]

def longest_palind(str):
    n = len(str)
    result = ''
    for i in range(n):
        for j in range(i, n):
            if str[i] == str[j]:
                s = str[i:j+1]
                if (is_palindrome(s) and len(s) > len(result)):
                    result = s
    return len(result), result

try:
    str = input()
except EOFError:
    str = ''

print(*longest_palind(str))
```

Nhóm làm tốt, đúng yêu cầu của đề

3. Nhóm 3:

```
def kiemtra(s):
    for i in range(len(s),0,-1):
        m=0
        while(m+i<=len(s)):
            temp=s[m:m+i]
            temp1=temp[::-1]
            if(temp==temp1):
                print(i,temp)
                return
            else:
                m=m+1
    print(0)
    return
try:
    s = input()
    s=s.strip()
    kiemtra(s)
except EOFError:
    print(0)
```

Nhóm làm tốt, đáp ứng yêu cầu của đề và đạt điểm tối đa.

4. Nhóm 4:

```
def longestPS(str):
    n = len(str)
    maxLength = 1
    start = 0
    for i in range(n):
        for j in range(i, n):
            flag = 1
            for k in range(0, ((j - i) // 2) + 1):
                if (str[i + k] != str[j - k]):
                    flag = 0
            if (flag != 0 and (j - i + 1) > maxLength):
                start = i
                maxLength = j - i + 1
    print(maxLength,str[start:start+maxLength] )
if __name__ == '__main__':
  try:
    str = input()
    longestPS(str)
  except E0Ferror:
    print(0)
```

Nhóm sử dụng phương pháp BF, tuy nhiên do chưa tối ưu ở chỗ tìm chuỗi đảo ngược, kiểm tra không thỏa không break ra nên chưa đạt điểm tối đa.

```
def longestPS(str):
  n = len(str)
  if n == 1:
    print(n,str)
  else:
    maxLength = 1
    for i in range(1,n):
        high=i+1
        low=i-1
        while low>=0 and high<n and str[low]==str[high]:
          if high - low +1 >maxLength:
            start=low
            maxLength = high - low + 1
          low= low -1
          high=high+1
        high = i+1
        low = i
        while low>=0 and high<n and str[low]==str[high]:
          if high - low +1 >maxLength:
            start=low
            maxLength = high - low + 1
          low= low -1
          high=high+1
    print(maxLength,str[start:start+maxLength] )
if __name__ == '__main__':
  try:
    str = input()
    longestPS(str)
  except EOFError:
    print(0)
```

Sau đó, nhóm chuyển qua phương pháp khác. Mặc dù điểm tối đa nhưng không đáp ứng yêu cầu của đề.

5. Nhóm 5:

```
S = input()
if len(S) == 0:
    exit(0)

def checkPalindrome(s):
    for i in range(len(s) // 2):
        if s[i] != s[len(s) - i - 1]:
            return False
    return True

for pad in range(len(S), 0, -1):
    for start in range(0, len(S) - pad + 1):
        subString = S[start:start+pad]
        if checkPalindrome(subString) == True:
            print(len(subString), subString)
            exit(0)
```

Nhóm làm đúng phương pháp, tuy nhiên vì kiểm tra trường hợp chuỗi rỗng bị sai nên chưa đạt điểm tối đa.

6. Nhóm 6:

```python
def printSubStr(str, low, high):
    for i in range(low, high + 1):
        print(str[i], end = "")

def longestPalSubstr(str):
    n = len(str)
    maxLength = 1
    start = 0
    i=0
    while(i<= n-maxLength):
        for j in range(i+maxLength-1, n):
            flag = 1
            for k in range(0, ((j - i) // 2) + 1):
                if (str[i + k] != str[j - k]):
                    flag = 0
            if (flag != 0 and (j - i + 1) > maxLength):
                start = i
                maxLength = j - i + 1
        i+= 1
    print(maxLength, end = ' ')
    printSubStr(str, start, start + maxLength - 1)
    return maxLength

if __name__ == '__main__':
    str = input()
    if (len(str)==0):
        print('0')
    else:
        longestPalSubstr(str)
```

Đầu tiên, nhóm sử dụng phương pháp Brute Force để giải quyết, tuy nhiên do chưa tối ưu ở chỗ tìm chuỗi đảo ngược, kiểm tra không thỏa không break ra nên bị time limit exceeded.

```python
def printSubStr(str, low, high):
    for i in range(low, high + 1):
        print(str[i], end = "")

def longestPalSubstr(str):
    n = len(str)
    maxLength = 1
    start = 0
    mid = 1
    while(mid<= n-maxLength):
        low = mid-1
        high = mid
        if (str[low] != str[high]):
            high= mid+1
        while(low>=0 and high<n and str[low]==str[high]):
            low-= 1
            high+= 1
        low+=1
        high-=1
        if (high-low+1>maxLength):
            start = low
            maxLength = high-low+1
        mid+=1
    print(maxLength, end = ' ')
    printSubStr(str, start, start + maxLength - 1)
    return maxLength

if __name__ == '__main__':
    try:
        str = input()
        longestPalSubstr(str)
    except EOFError:
        print(0)
```

Sau đó nhóm có tối ưu và đã đạt điểm tối đa, tuy nhiên lại không dùng Brute Force.

7. Nhóm 7:

```python
def isPal(s):
    return s == s[::-1]
try:
    s = input()
    result = ''

    for i in range(len(s)):
        key  = False
        temp = ''
        for j in range(len(s) - 1, i - 1, -1):
            if(s[i] == s[j]):
                temp = s[i: j + 1]
                if isPal(temp) and len(temp) > len(result):
                    key = True
                    result = temp
                    break
except EOFError as error:
    # Output expected EOFErrors.
    Logging.log_exception(error)

print(str(len(result)) + ' ' + result)
```

Nhóm sử dụng phương pháp BF, tuy nhiên chưa kiểm tra trường hợp chuỗi rỗng nên chưa đạt điểm tối đa.

8. Nhóm 8:

```python
def longestPalSubstr(string):
    maxLength = 1
    start = 0
    length = len(string)
    low = 0
    high = 0
    for i in range(1, length):
        low = i - 1
        high = i
        while low >= 0 and high < length and string[low] == string[high]:
            if high - low + 1 > maxLength:
                start = low
                maxLength = high - low + 1
            low -= 1
            high += 1

        low = i - 1
        high = i + 1
        while (low >= 0) and (high < length) and (string[low] == string[high]):
            if high - low + 1 > maxLength:
                start = low
                maxLength = high - low + 1
            low -= 1
            high += 1
    print (maxLength, string[start:start + maxLength])
try:
    string = input()
    longestPalSubstr(string)
except EOFError:
    print(0)
```

Nhóm không sử dụng phương pháp thiết kế BF, tuy đạt điểm tối đa nhưng không đáp ứng yêu cầu của đề. Cũng giống như nhóm các nhóm trước, những lần nộp đầu vẫn không kiểm tra trường hợp chuỗi rỗng.

9. Nhóm 11:

```
def DX(s):
    i = 0
    j = len(s) - 1
    while (i < j):
        if s[i] != s[j]: return False
        i += 1
        j -= 1
    return True
try:
    ss = input().strip()
    t = DX(ss)
    lenth = len(ss)
    maxx = 0
    ts = ''
    for i in range(lenth - 1):
        for j in range(lenth - 1,i,-1):
            s = ss[i:j+1]
            if DX(s) == True and maxx < len(s):
                maxx = len(s)
                ts = s
    if lenth > 0 and maxx == 0:
        print(1,ss[0])
    else:
        print(maxx,ts)
except:
    print(0)
```

Nhóm làm đúng phương pháp, đạt điểm tối đa.

10. Nhóm 12:

```
1  import sys
2  S = sys.stdin.readline()
3  if len(S) == 0:
4      print(0, '')
5      exit(0)
6
7  def checkPalindrome(s):
8      for i in range(len(s) // 2):
9          if s[i] != s[len(s) - i - 1]:
10             return False
11     return True
12
13 for pad in range(len(S), 0, -1):
14     for start in range(0, len(S) - pad + 1):
15         subString = S[start:start+pad]
16         if checkPalindrome(subString) == True:
17             print(len(subString), subString)
18             exit(0)
```

Nhóm làm tốt, đáp ứng yêu cầu của đề.

Nhận xét:

- Các nhóm nộp bài đầy đủ
- Tuy một số nhóm không vượt qua được testcase chuỗi rỗng nên không đạt điểm tối đa, nhưng do đây là testcase bẫy nên không có vấn đề gì
- Một số nhóm bị time limit exceeded là do chưa tối ưu ở quá trình kiểm tra chuỗi đối xứng, tuy nhiên vẫn vận dụng được phương pháp Brute Force
- Các nhóm về cơ bản nắm được cách vận dụng phương pháp thiết kế thuật toán Brute Force vào giải bài toán cụ thể qua bài tập ở lớp và bài tập về nhà.