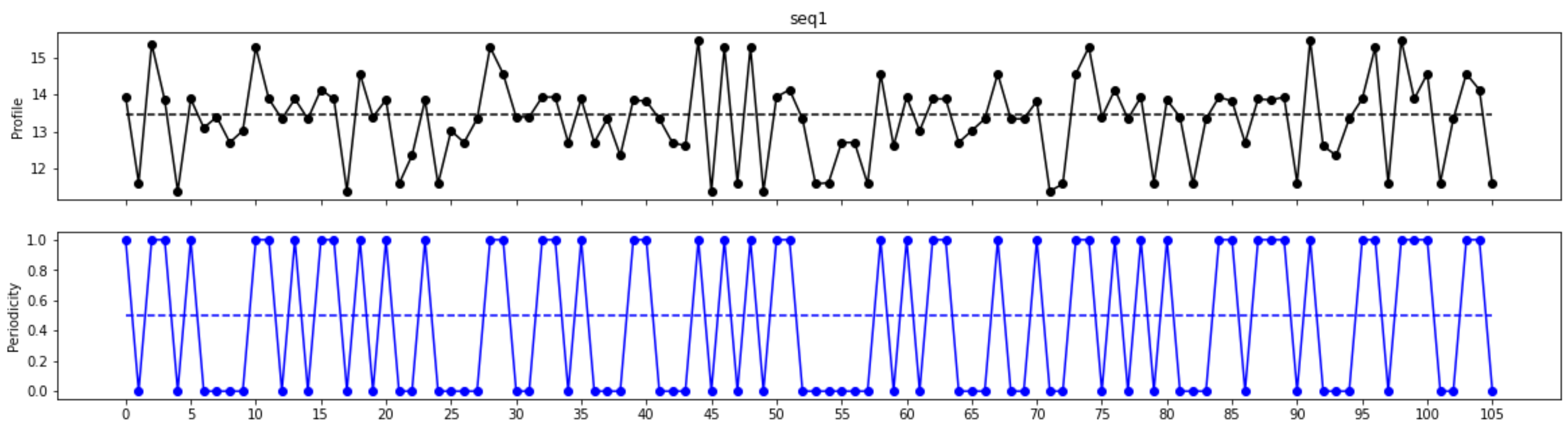
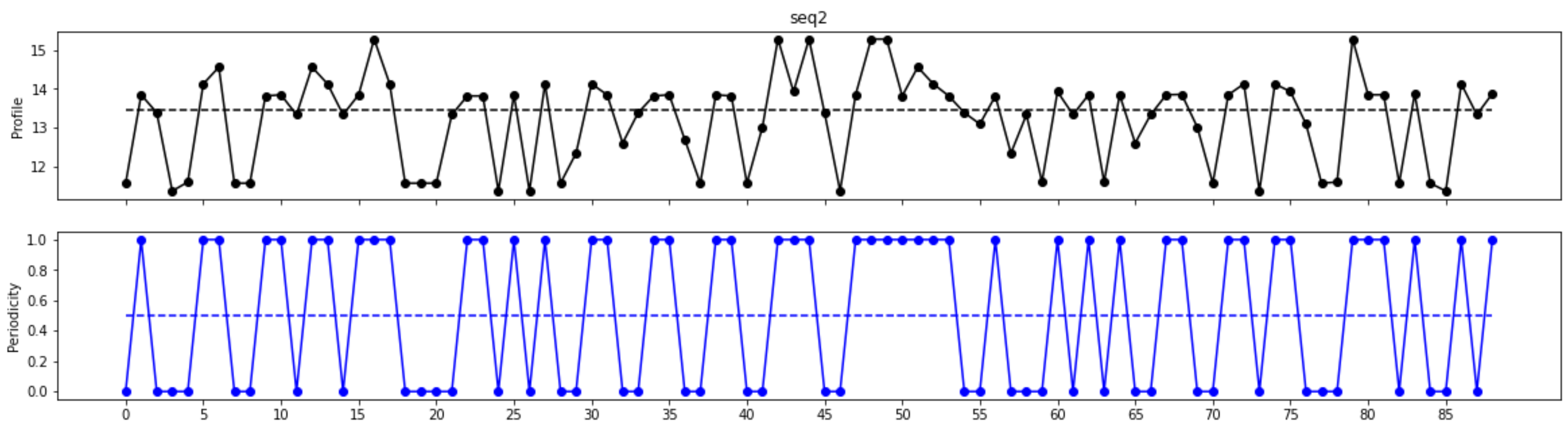
Question 1

In [1]:
```
from DnaUtils import readFasta
import matplotlib.pyplot as plt
import pandas as pd
pd.options.display.max_rows = 999
import re
import numpy as np
ids,seqs = readFasta("Q1.fasta")
s = "A: 13.85 D: 11.61 C: 15.37 E: 11.38 F: 13.93 G: 13.34 H: 13.82 I: 15.28 K: 11.58 L: 14.13 \
M: 13.86 N: 13.02 P: 12.35 Q: 12.61 R: 13.10 S: 13.39 T: 12.70 V: 14.56 W: 15.48 Y: 13.88"
hydro = dict(zip([x[0] for x in s.split(' ')[::2]],map(float,s.split(' ')[1::2])))
calc = lambda x : hydro[x]
binary = lambda x : int(hydro[x] > mean)
```

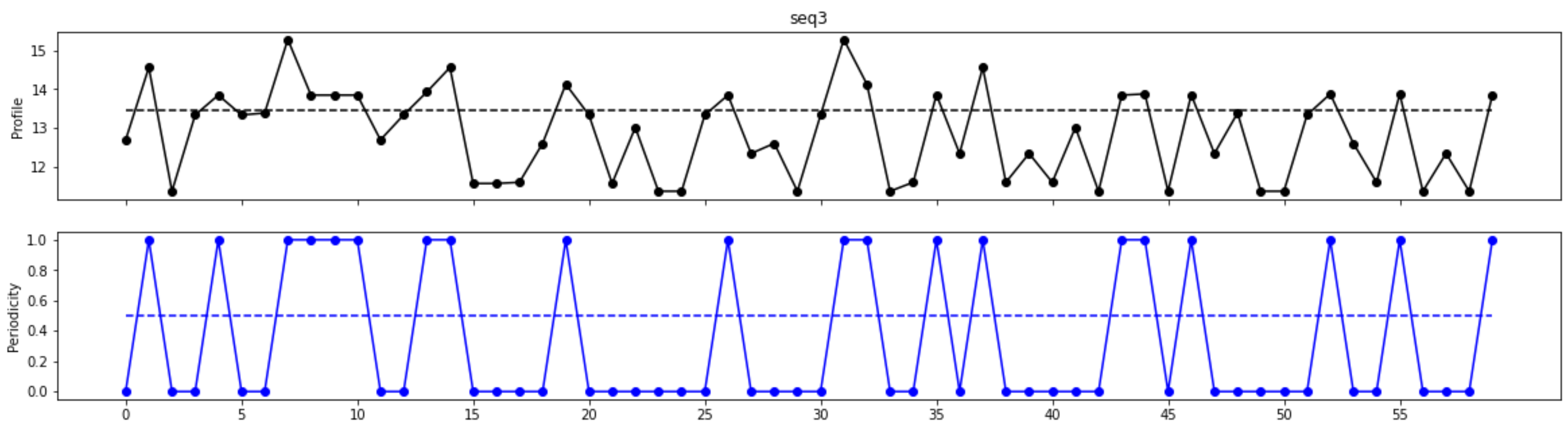
In [2]:
```
aseqs = []
bseqs = []
for i,seq in zip(ids,seqs):
    mean = np.mean(list(hydro.values()))
    fig,axs = plt.subplots(2,1,sharex=True, figsize = (20,5))
    axs[0].set_title(i[1:])
    axs[0].set_ylabel("Profile")
    axs[0].plot(list(map(calc,seq)), '-ok')
    axs[0].plot([mean]*len(seq), '--k')
    axs[1].plot(list(map(binary, seq)), '-ob')
    axs[1].plot([0.5]*len(seq), '--b')
    axs[1].set_ylabel("Periodicity")
    axs[1].set_xticks(range(0,len(seq),5))
    plt.show()
    bin_seq = (''.join(map(str,(map(binary, seq)))))
    helices = re.finditer('(0011){2,}|(1100){2,}',bin_seq)
    sheets = re.finditer('(01){3,}|(10){3,}',bin_seq)
    helices = [f'{x.start()}-{x.end()}' for x in list(helices)]
    sheets = [f'{x.start()}-{x.end()}' for x in list(sheets)]
    helix_seqs = [seq[int(x.split('-')[0]):int(x.split('-')[1])] for x in helices if len(x) > 0 ]
    sheet_seqs = [seq[int(x.split('-')[0]):int(x.split('-')[1])] for x in sheets if len(x) > 0 ]
    aseqs.append(helix_seqs)
    bseqs.append(sheet_seqs)
    print("Alpha Helices:")
    print(f"Positions : {helices}")
    print(f"Sequences {helix_seqs}\n")
    print("Beta Sheets:")
    print(f"Positions : {sheets}")
    print(f"Sequences {sheet_seqs}\n")
```



```
Alpha Helices:
Positions : ['26-34']
Sequences ['TGIVSSFF']

Beta Sheets:
Positions : ['16-22', '43-51', '57-63', '74-82']
Sequences ['YEVSMK', 'QWEIDIEF', 'KVQFNY', 'ISLGFDAS']
```



```
Alpha Helices:
Positions : ['3-11', '28-44', '65-73']
Sequences ['EDLVKKHA', 'KPLAQSHATKAHKNIF', 'QGAMNKAL']

Beta Sheets:
Positions : ['23-29', '59-65']
Sequences ['HEAELK', 'DFGADA']
```



```
Alpha Helices:
Positions : ['9-17']
Sequences ['AATGFVKK']

Beta Sheets:
Positions : []
Sequences []
```

Question 2

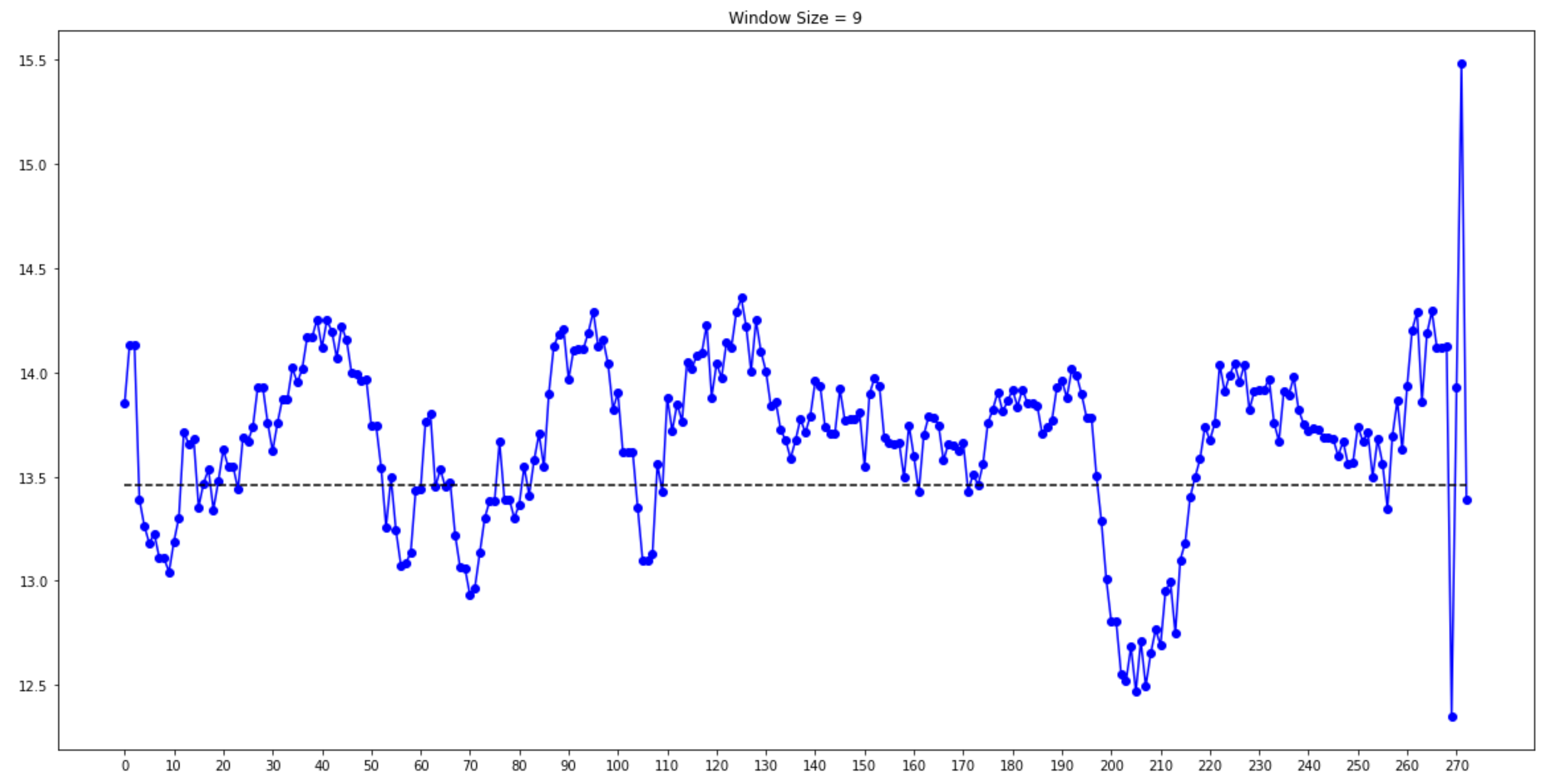
To calculate the amphiphatic index, we take absolute difference between the average hydrophobicity values above and below the dotted black line in the above graphs, for alpha helices and beta sheets

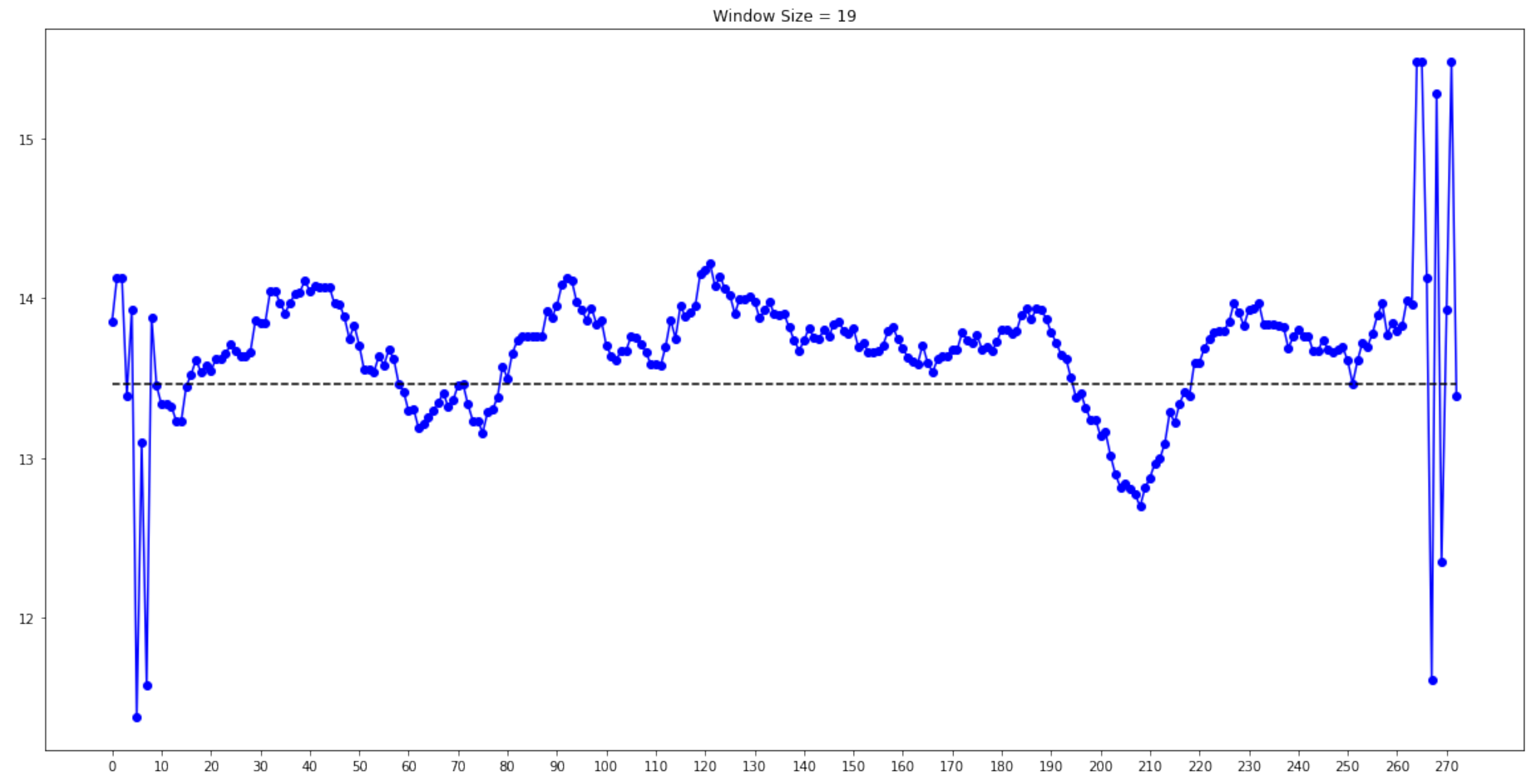
In [3]:
```
for i,seq,alpha,beta in zip(ids,seqs,aseqs,bseqs):
    print(i)
    print("1. Alpha Helix")
    for helix in alpha:
        for i in range(len(helix)-7):
            temp = np.resize(list(map(calc,helix[i:i+8])),(4,2))
            print(f"{helix[i:i+8]} : {abs(np.round(np.mean(temp[::2].flatten())-np.mean(temp[1::2].flatten()),3))}
    print()
    print("2. Beta Sheet")
    for sheet in beta:
        for i in range(len(sheet)-5):
            temp = list(map(calc,sheet[i:i+6]))
            print(f"{sheet[i:i+6]} : {np.round(abs(np.mean(temp[::2])-np.mean(temp[1::2])),3)}")
    print("------------------------\n")
```

```
>seq1
1. Alpha Helix
TGIVSSFF : 1.22

2. Beta Sheet
YEVSMK : 1.983
QWEIDI : 3.48
WEIDIE : 3.89
EIDIEF : 3.373
KVQFNY : 1.72
ISLGFD : 1.667
SLGFDA : 1.19
LGFDAS : 1.19
------------------------

>seq2
1. Alpha Helix
EDLVKKHA : 2.553
KPLAQSHA : 1.43
PLAQSHAT : 0.17
LAQSHATK : 1.342
AQSHATKA : 0.093
QSHATKAH : 1.265
SHATKAHK : 0.173
HATKAHKN : 1.615
ATKAHKNI : 0.445
TKAHKNIF : 2.0
QGAMNKAL : 1.285

2. Beta Sheet
HEAELK : 2.487
DFGADA : 1.69
------------------------

>seq3
1. Alpha Helix
AATGFVKK : 1.748

2. Beta Sheet
------------------------
```

Question 3

In [4]:
```
ids, (seq,) = readFasta("Q2.fasta")
temp = (list(map(calc,seq)))
def rolling_window(seq,window):
    temp = (list(map(calc,seq)))
    n = window//2
    return [np.mean(temp[i-n:i+n+1]) if n <= i <= len(seq)-n-1 else temp[i] for i in range(len(seq))]
for window in [9,19]:
    plt.figure(figsize = (20,10))
    plt.plot(rolling_window(seq,window),'o-b')
    mean = np.mean(list(hydro.values()))
    plt.plot([mean]*len(seq), '--k')
    plt.title(f'Window Size = {window}')
    plt.xticks(range(0,len(seq),10))
    plt.show()
```





Transmembrane segments are highly hydrophobic, so from the graphs above, regions with high hydrophobicity are likely to be hydrophobic.

```
In [5]:  print(f"20 to 50 : {seq[20:50]}")
         print(f"82 to 100 : {seq[82:100]}")
         print(f"115 to 130 : {seq[115:130]}")
         print(f"225 to 240 : {seq[225:240]}")
```

```
20 to 50 : LFDFWVGPYFVGFFGVSAIFFIFLGVSLIG
82 to 100 : GGFWQAITVCALGAFISW
115 to 130 : HVPLAFCVPIFMFCV
225 to 240 : ALSIHRLGLFLASNI
```

Question 4

1. Pattern 1 : 10406 hits in 10000 sequences
2. Pattern 2 : 3814 hits in 3751 sequences

Question 5

In [6]:

```
ids, seqs = readFasta("Q4.fasta")
p1, p2 = "[SV]-T-[VT]-[DERK](2)-{IL}", "[FILV]Qxxx{RK}Gxxx[RK]xx[FILVWY]"
def convert_to_regex(pattern):
    pattern = pattern.replace('{','[^')
    pattern = pattern.replace('}',']')
    pattern = pattern.replace('(','{')
    pattern = pattern.replace(')','}')
    pattern = pattern.replace('x','.')
    pattern = pattern.replace('-','')
    return pattern
data = []
used = set()
for i,seq in zip(ids,seqs):
    if i not in used:
        temp = list(re.finditer(convert_to_regex(p1),seq))
        if temp:
            for x in temp:
                data.extend([[i.split('|')[0][1:],x.start(),x.end()]])
        used.add(i)
df = pd.DataFrame(data, columns = ['id','start','stop'])
df.index = df.id
df.drop(['id'],axis = 1, inplace = True)
```

In [7]:

```
df
```

Out[7]:

```
          start  stop
      id
  4A0C_2    664   670
  4A0K_1    665   671
  5F0J_3     69    75
  5F0L_3     69    75
  5F0M_3     69    75
  5F0P_3     69    75
  5N69_1     68    74
  5N6A_1     68    74
  5TBY_1     68    74
  6FSA_1     68    74
  6X5Z_3     68    74
  7JH7_2     68    74
  1A8J_1    203   209
  1ADQ_2    200   206
  1AIV_1    542   548
  1AQK_1    203   209
  1BJM_1    203   209
  1DCL_1    203   209
  1DPU_1     85    91
  1E8I_1     18    24
  1FM2_2    111   117
  1FM5_1     39    45
 1FNT_15    152   158
  1FO9_1     18    24
  1FOA_1     18    24
  1GPJ_1    315   321
  1IQ7_1    201   207
  1JT1_1    121   127
  1JVK_1    204   210
  1JVZ_2    111   117
  1K07_1    121   127
  1KEH_1    280   286
  1KVD_2     45    51
  1KVE_2     45    51
```

1L9Y_1	121	127
1LGV_1	204	210
1LHZ_1	204	210
1LIL_1	199	205
1MCB_1	203	209
1MCC_1	203	209
1MCE_1	203	209
1MCF_1	203	209
1MCH_1	203	209
1MCJ_1	203	209
1MCL_1	203	209
1MCO_1	203	209
1MCR_1	203	209
1MCS_1	203	209
1NL0_1	203	209
1OVT_1	542	548
1Q1J_1	204	210
1R5M_1	11	17
1RP1_1	436	442
1RYX_1	542	548
1RZF_1	203	209
1S5J_1	603	609
1UYP_1	370	376
1VQT_1	71	77
1W72_5	201	207
1WF5_1	28	34
1Z1D_1	89	95
1Z7Q_15	152	158
1ZTM_1	227	233
1ZVO_1	201	207
2APC_1	12	18
2B0S_1	205	211
2B1A_1	205	211
2B1H_1	205	211
2BB0_1	113	119
2BB5_1	334	340
2DD8_2	200	206
2DVV_1	53	59
2E3K_1	53	59
2E7N_1	60	66
2ES7_1	24	30
2FB4_1	203	209
2FH6_1	521	527
2FH8_1	521	527
2FHB_1	521	527
2FHC_1	521	527
2FHF_1	521	527
2FL5_1	199	205
2G3F_1	113	119
2G4A_1	49	55
2G75_2	200	206
2GAN_1	10	16

2H32_2	108	114
2H3N_2	107	113
2IDR_1	34	40
2IDV_1	34	40
2IG2_1	203	209
2J28_23	51	57
2J42_1	337	343
2J6E_3	222	228
2JB5_2	204	210
2JB6_1	204	210
2JE8_1	348	354
2JE8_1	474	480
2JQ3_1	54	60
2KC8_1	22	28
2KC9_1	22	28
2LDX_1	0	6
2MCG_1	203	209
2MXC_1	74	80
2OAJ_1	752	758
2OAJ_1	824	830
2OLD_1	204	210
2OMB_1	204	210
2OMN_1	204	210
2PI2_1	256	262
2QA2_1	165	171
2RCJ_2	201	207
2RDO_14	51	57
2RDO_34	56	62
2VJX_1	346	352
2VJX_1	472	478
2VL4_1	346	352
2VL4_1	472	478
2VMF_1	346	352
2VMF_1	472	478
2VO5_1	346	352
2VO5_1	472	478
2VOT_1	346	352
2VOT_1	472	478
2VQT_1	346	352
2VQT_1	472	478
2VQU_1	346	352
2VQU_1	472	478
2VR4_1	346	352
2VR4_1	472	478
1AA0_1	40	46
1FO8_1	13	19
1JKM_1	344	350
1JW0_2	111	117
1MCD_1	203	209
1MCI_1	203	209
1MCK_1	203	209
1MCN_1	203	209

Question 6

-->20 sequences were loaded from uniport by searching "Beta barrel"

In [48]:

```python
ids, seqs = readFasta("Q6.fasta")
ids = [x.split('|')[1] for x in ids]
df = pd.DataFrame(index = ids)
df["Server1"] = [1,0,0,0,1,0,0,0,0,1]
df["Server2"] = [1,0,0,0,1,0,0,0,0,1]
print("Number of matches for each server")
df
```

```
Number of matches for each server
```

Out[48]:

	Server1	Server2
Q6UD73	1	1
P06996	0	0
P0A910	0	0
O18423	0	0
P0A940	1	1
P66948	0	0
P9WIU5	0	0
Q1I8U1	0	0
Q2MJ20	0	0
H1A981	1	1