

Programma python

Scrivere un programma python che:

- prenda in ingresso un file di allineamenti multipli con una sequenza di riferimento
- stampi la sequenza di riferimento, una sequenza “consenso” con in ogni posizione il residuo più frequente nell’allineamento, e l’indicazione se la sequenza consenso ha una delezione (D), una modifica (M) o è conservata (C) rispetto alla sequenza di riferimento

Esempio file

```
0.012 C -----CCAHSLVVTEASAD-----
0.013 G -----GGGYFAVFATAFSS-----
0.024 V -----VILINLLLYVIIIE-----VI-----
0.014 P -----PPPCKRKGK-SEDS-----RV-----
0.013 A -----AKACLSSVY-VIN-----FT-----L-----
0.014 I -----IIPPEY-QLLVFI-----HL-----P-----
0.019 Q -----QEKENEETGGGE-----LG-----L-----
0.020 P -----PPTPPSSQKP-DI-----SV-----P-----A-----
0.077 V --N-----VKAP-PPRESTPVGAVKGTCT--P--L--SR-E---L-DC---AE-W-S---MQ-
0.087 L --A-----LLLP-VISESPKEDHSPPSSP-----L--E---FD-IG---WDRG---EA-T-T---QI-
0.106 S --K-----SPLL-GSEERIPVTSSSGTGF-----N--KA--ADDQT-F-RLW--KHP-G-P---TQ-
0.103 G --T-----GGQY-V-P IQKPDNIRFFQTSF-----DG-LT--DSRGA-Y-TDSD--AVD-R-Q---VA-
0.202 L --F-----LAA--RQLQSDSSGPAEMSHG-QQQSSSRD-SGQSKTRNANV-HQMPV-LRGLH-Q--NEGS
0.274 S VVI-E-PPTSNE--EFYSSNVLLPKGGGSA-GGGFLSEGEENPNSDSGR-ASASGPKSFAP-P-DENQT
0.399 R GGR-R-RRRRRR-QKRRRKR-RFRRTRR-RRRATSRKKRERGMRRERS-RRLRDELETR-SAALKDW
0.610 I IVI-I-IIIIIM-IIVIVIIIVIVIRII-IIIIIIIIIIIIIVIIIIII-IIIIVHYVITI-IYLVIVL
0.463 V YAV-V-VIVVVAVLLIVIVVVIVIVIV-LMAHQRVVVIVYAVRVVVV-IIIIVAINMVVYPISHLRII
...
```

Esempio esecuzione

```
> python consensus.py
inserire nome file: alignment
C - D
G - D
V - D
P - D
A - D
I - D
Q - D
P - D
V - D
L - D
S - D
G - D
L - D
S S C
R R C
I I C
V V C
N G M
G G C
E S M
...
```

Programma python: suggerimento

Si possono implementare 5 funzioni separate:

1. Una che legga il file e restituisca la sequenza di riferimento e una lista di profili (mappe residuo → numero di occorrenze nell’allineamento)
2. Una che dato la lista di profili restituisca la sequenza consenso
3. Una che date la sequenza di riferimento e la sequenza consenso, restituisca la sequenza di cambiamenti (‘M’ per modifica, ‘D’ per delezione, ‘C’ per conservazione)

4. Una che stampi le sequenze di riferimento, consenso e di cambiamenti appaiate
5. una che realizzi il programma richiesto usando le funzioni di cui sopra

Shell: esercizio #1

Nel file `sequences.fasta`, le intestazioni delle sequenze indicano la localizzazione subcellulare della proteina, es.

```
>7B2_HUMAN:Secretory
```

Calcolare quante sequenze residenti nel nucleo “Nucleus” contengono un motivo “*receptor box*”, identificato dalla espressione regolare

$$[^P]L[^P][^P]LL[^P]$$

È preferibile, ma non necessario, risolvere l’esercizio in una sola riga di comando.

Risultato atteso

337.

Shell: esercizio #2

Dato il file `sequences.fasta` ed i due seguenti motivi:

- Al massimo una metionina (M); seguita da una arginina (R) o da una lisina (K); seguita da un aa. qualunque che non sia una prolina (P). Il motivo deve trovarsi all’**inizio** della sequenza.
- Una lisina; seguita da al massimo un aa. qualunque; seguito da una lisina; seguita da due o tre aa. qualunque. Il motivo deve trovarsi alla **fine** della sequenza.

Calcolare quante sequenze contengono il primo motivo, quante il secondo, e quante entrambi, cercando di minimizzare il numero di invocazioni di `grep`.

Risultato atteso

308, 93, 391.

Modalità di esecuzione e consegna

1. Avviare la macchina in modalità ESAME
2. Autenticarsi con nome utente `sci-esame` e password fornita dal docente
3. Il testo del compito ed i file necessari si trovano in una cartella `Testo` sul Desktop
4. Realizzare il programma python come file `programma.py` e scrivere gli esercizi da linea di comando in un file di testo `linea_di_comando.txt`
5. Creare sul Desktop una cartella con `nome_cognome` e metterci i due file realizzati.
6. Eseguire il logout ma NON spegnere la macchina