Warunki jakie powinno spełniać rozwiązanie

1. W implementacji można korzystać tylko z elementarnych konstrukcji Python'a, takich jak: funkcje, rekurencja instrukcje warunkowe, pętle.

2. Dopuszczalne jest oczywiście implementowanie dodatkowych funkcji pomocniczych.

Format rozwiązań

Implementacja funkcji powinna się znajdować w pliku o nazwie `zad1.py`. Krótki opis rozwiązania powinien być umieszczony na początku pliku w formie komentarza (w pierwszej linii w komentarzu powinno być imię i nazwisko studenta). Niedopuszczalne jest w szczególności:

1. zmienianie nazwy funkcji implementującej algorytm, listy jej argumentów, lub nazwy pliku z rozwiązaniem,

2. wypisywanie na ekranie jakichkolwiek napisów, ew. napisy dodane na potrzeby diagnozowania błędów należy usunąć przed wysłaniem zadania.

Zadania niezgodne z powyższymi ograniczeniami otrzymają ocenę 0 pkt. Rozwiązania w innych formatach (np. `.PDF`, `.DOC`, `.PNG`, `.JPG`) z definicji nie będą sprawdzane i otrzymają ocenę 0 punktów, nawet jeśli będą poprawne.

Czas na rozwiązanie zadania 25 minut.
Oceniane będą: czytelność, poprawność i efektywność rozwiązań.

Zadanie 1

Dana jest tablica T zawierająca liczby wymierne reprezentowane w postaci ułamków. Ułamki reprezentowane są w postaci krotek składających się z licznika i mianownika. Proszę napisać funkcję `longest(T)`, zwracającą długość najdłuższego spójnego podciągu, którego elementy stanowią ciąg geometryczny. W przypadku gdy w tablicy nie ma ciągu dłuższego niż 2 elementy, funkcja powinna zwrócić wartość 0.

Komentarz: Można założyć, że tablica wejściowa liczy więcej niż 2 elementy.

Przykłady:

```
print(longest( [(0,2),(1,2),(2,2),(4,2),(4,1),(5,1)] ) # wypisze 4
print(longest( [(1,2),(-1,2),(1,2),(1,2),(1,3),(1,2)] ) # wypisze 3
print(longest( [(3,18),(-1,6),(7,42),(-1,6),(5,30),(-1,6)] ) # wypisze 6
print(longest( [(1,2),(2,3),(3,4),(4,5),(5,6)] ) # wypisze 0
```

Warunki jakie powinno spełniać rozwiązanie

1. W implementacji można korzystać tylko z elementarnych konstrukcji Python'a, takich jak: funkcje, rekurencja instrukcje warunkowe, pętle.

2. Dopuszczalne jest oczywiście implementowanie dodatkowych funkcji pomocniczych.

Format rozwiązań

Implementacja funkcji powinna się znajdować w pliku o nazwie `zad2.py`. Krótki opis rozwiązania powinien być umieszczony na początku pliku w formie komentarza (w pierwszej linii w komentarzu powinno być imię i nazwisko studenta). Niedopuszczalne jest w szczególności:

1. zmienianie nazwy funkcji implementującej algorytm, listy jej argumentów, lub nazwy pliku z rozwiązaniem,

2. wypisywanie na ekranie jakichkolwiek napisów, ew. napisy dodane na potrzeby diagnozowania błędów należy usunąć przed wysłaniem zadania.

Zadania niezgodne z powyższymi ograniczeniami otrzymają ocenę 0 pkt. Rozwiązania w innych formatach (np. `.PDF`, `.DOC`, `.PNG`, `.JPG`) z definicji nie będą sprawdzane i otrzymają ocenę 0 punktów, nawet jeśli będą poprawne.

Czas na rozwiązanie zadania 25 minut.
Oceniane będą: czytelność, poprawność i efektywność rozwiązań.

Zadanie 2

Dane jest słowo składające się z liter alfabetu angielskiego. Słowo to tniemy na kawałki, tak aby każdy kawałek zawierał dokładnie jedną samogłoskę. Proszę napisać funkcję `cutting(s)`, która zwraca liczbę sposobów pocięcia słowa na kawałki.

Przykłady:

```
print(cutting('student')) # wypisze 2 bo stu-dent, stud-ent
print(cutting('sesja')) # wypisze 3 bo se-sja, ses-ja, sesj-a
print(cutting('ocena')) # wypisze 4 bo o-ce-na, o-cen-a, oc-e-na, oc-en-a,
print(cutting('informatyka')) # wypisze 36
```

Warunki jakie powinno spełniać rozwiązanie

1. W implementacji można korzystać tylko z elementarnych konstrukcji Python'a, takich jak: funkcje, rekurencja instrukcje warunkowe, pętle.

2. Dopuszczalne jest oczywiście implementowanie dodatkowych funkcji pomocniczych.

Format rozwiązań

Implementacja funkcji powinna się znajdować w pliku o nazwie `zad3.py`. Krótki opis rozwiązania powinien być umieszczony na początku pliku w formie komentarza (w pierwszej linii w komentarzu powinno być imię i nazwisko studenta). Niedopuszczalne jest w szczególności:

1. zmienianie nazwy funkcji implementującej algorytm, listy jej argumentów, lub nazwy pliku z rozwiązaniem,

2. wypisywanie na ekranie jakichkolwiek napisów, ew. napisy dodane na potrzeby diagnozowania błędów należy usunąć przed wysłaniem zadania.

Zadania niezgodne z powyższymi ograniczeniami otrzymają ocenę 0 pkt. Rozwiązania w innych formatach (np. `.PDF`, `.DOC`, `.PNG`, `.JPG`) z definicji nie będą sprawdzane i otrzymają ocenę 0 punktów, nawet jeśli będą poprawne.

Czas na rozwiązanie zadania 25 minut.
Oceniane będą: czytelność, poprawność i efektywność rozwiązań.

Zadanie 3

Dana jest definicja klasy, której obiekty stanowią elementy listy odsyłaczowej:

```
class Node:
..def init(self,val,next=None):
....self.val = val
....self.next = next
```

Lista zawierała wartości stanowiące kolejne wyrazy ciągu arytmetycznego. Z wnętrza listy usunięto pewną liczbę elementów. Proszę napisać funkcję `repair(p)`, (p wskazuje na pierwszy element listy) która uzupełnia listę elementami, tak aby ponownie zawierała kolejne wyrazy ciągu arytmetycznego. Funkcja powinna zwrócić liczbę wstawionych elementów.

Komentarz: Można założyć, że lista wejściowa liczy więcej niż 2 elementy.