PPPD - Lab. 07

Zadanie punktowane, lab 07, grupa A, 2021/2022, autor: Piotr Wolszakiewicz

Temat: Zabawy na listach

UWAGA: Nie wolno korzystać z append i slice

Treść zadania

Zadanie będzie składało się z kilku etapów. Będziemy w nim korzystali z kodowania znaków w systemie ASCII. Każdy znak w kodowaniu ASCII ma przypisaną wartość dziesiętną z zakresu 0 do 255. np. małe litery z zakresu a-z mają kolejne wartości pomiędzy 97-122 w sumie 26 znaków

```
a - 97
b - 98
...
y - 121
z - 122
```

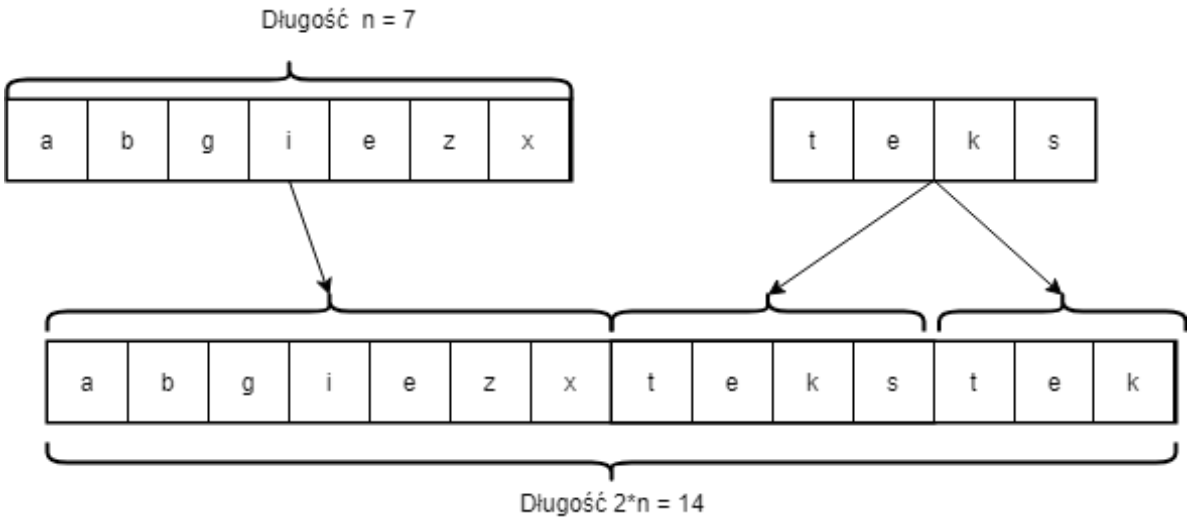
Aby zamienić znak na wartość dziesiętną, możemy użyć funkcji `ord`, aby zamienić wartość dziesiętną na znak używamy funkcji `chr`

```
ord('a')   # zwróci wartość 97
chr(97)    # zwróci wartość 'a'
```

Na początku trzeba poprosić użytkownika o wprowadzenie liczby dziesiętnej z zakresu [5, 15] - będzie to nasza bazowa długość `-dlugosc`. Podanie niepoprawnej wartości powinno skutkować wyrzuceniem błędu.

W ramach programu należy napisać funkcje zgodnie z poniższym opisem. Ich wywołanie powinno się znaleźć w głównej funkcji programu `main()`:

- `losuj_litery` - funkcja przyjmuje jako argument liczbę całkowitą `dlugosc`. Zwraca listę o długości `dlugosc` z losowo wygenerowaymi literami od `a` do `z`.
  Ziarno generatora liczb pseudolosowych nalezy ustawić w funkcji `main()` na `2014`.
- `polacz_z_zawijaniem` - funkcja przyjmuje jako argumenty dwie listy `pierwsza` i `druga`. Jako wynik zwraca listę o wielkości dwa razy większej, niż większa z list. Wartości w liście są przepisane z oryginalnych list z zawinięciem tzn. krótsza lista jest powtarzana do wypełnienia swojej połowy. Pierwsza lista jest przepisywana w pierwszą połówkę, druga w drugą.




```
    pierwsza = 'kr'
    druga = 'dluga'
    połączona = polacz_z_zawijaniem(pierwsza, druga)
    # Powinniśmy otrzymać listę z znakami:
    # ['k','r','k','r', 'k','d','l','u','g','a']
```

- `odwroc` - funkcja przyjmuje jako argumenty listę, oraz indeks początkowy i końcowy. Funkcja zamienia kolejność elementów w liście w taki sposób, że pierwszy element zamienia się z ostanim, drugi z przedostatnim itd. Zamiana elementow odbywa się jednak w przedziale pomiędzy `indeks_startu` i `indeks_konca`. Funkcja nic nie zwraca. Elementy powinny zostać zamienione w oryginalnej liście, bez używania pomocniczych list.

```
    a = ['d','l','u','g','a','k','r','k','r', 'k']
    odwroc(a, 1, 4)
    print(a)
    # ['d','a','g','u','l','k','r','k','r', 'k']
```

- `przesun_w_lewo` - funkcja przesuwa elementy listy w lewo z zawinięciem tzn. pierwszy element przesunięty poza listę staje się ostatnim. Funkcja jako argumenty przyjmuje listę, którą będziemy rotować i liczbę całkowitą określającą ile ma być rotacji. Funkcja nic nie zwraca, rotacja powinna zmieniać znaki w oryginalnej liście. Rotacja ma być zrealizowana bez użycia pomocniczych list.

```
a = ['t','a','b','l','i','c','a']
przesun_w_lewo(a, 3)
print(a)
# ['l','i','c','a','t','a','b']
```

- `mod_26` - funkcja przyjmuje jako argument listę `tab` ze znakami z zakresu od `a` do `z` i przekształca ją w taki sposób, że: element pod indeksem k będzie miał wartość (f(tab[k-1]) + f(tab[k]) + f(tab[k+1])) mod 26 - skonwertowaną na wartość znakową `a` - `z` (wartości do sumy pochodzą z oryginalnej listy), gdzie f - funkcja zamieniająca znak na wartość liczbową. Jeśli tab[k-1], bądź tab[k+1] nie istnieje trzeba przyjąć dla niej wartość 0. Znaki mają być zamienione w oryginalnej liście, bez użycia pomocniczych list, można natomiast użyć kilku pomocniczych zmiennych.

```python
m = ['a', 'b', 'c', 'd']
print(m)
mod_26(m)
print(m)
# ['n', 'i', 'l', 'r']
```

Przykłady interakcji użytkownika z programem

```
Podaj długość bazową: 7
losuj_litery: 7
['k', 's', 'm', 'g', 'k', 'g', 'n']
losuj_litery: 15
['n', 'w', 'p', 'a', 't', 'm', 'l', 'e', 'v', 'u', 's', 'o', 'f', 'p', 'g']
polacz_z_zawijaniem
['k', 's', 'm', 'g', 'k', 'g', 'n', 'k', 's', 'm', 'g', 'k', 'g', 'n', 'k',
'n', 'w', 'p', 'a', 't', 'm', 'l', 'e', 'v', 'u', 's', 'o', 'f', 'p', 'g']
odwroc pomiedzy 3, 7
['n', 'w', 'p', 'a', 't', 'm', 'l', 'e', 'v', 'u', 's', 'o', 'f', 'p', 'g']
['n', 'w', 'p', 'e', 'l', 'm', 't', 'a', 'v', 'u', 's', 'o', 'f', 'p', 'g']
przesun_w_lewo o 3
['k', 's', 'm', 'g', 'k', 'g', 'n']
['g', 'k', 'g', 'n', 'k', 's', 'm']
mod_26
['a', 'b', 'c', 'd']
['n', 'i', 'l', 'r']
```

Punktacja

Za poszczególne elementy można uzyskać następującą liczbę punktów:

- Prawidłowo stworzony rdzeń programu: obsługa wczytania bazowej dłubości z walidacją oraz funkcja `losuj_litery` - 1pkt
- Poprawnie zaimplementowana funkcja `polacz_z_zawijaniem` - 2pkt
- Poprawnie zaimplementowana funkcja `odwroc` - 2pkt
- Poprawnie zaimplementowana funkcja `przesun_w_lewo` - 2pkt
- Poprawnie zaimplementowana funkcja razem z przypadkami złośliwymi `mod_26` - 3pkt
- Każdy z wymienionych etapów wymaga, aby były stworzone odpowiednie funkcje do tego etapu i aby były one prawidłowo wywołane/przetestowane w `main`

Uwaga

- Jeśli program się nie kompiluje (interpretuje), ocena jest zmniejszana o połowę
- Jeśli kod programu jest niskiej jakości (nieestetycznie formatowanie, mylące nazwy zmiennych itp.), ocena jest zmniejszana o 2pkt
- Jeśli założenia odnośnie nieużywania dodatkowej listy nie są spełnione, punkty za dany etap są wyzerowane