

## Warunki jakie powinno spełniać rozwiązanie

1. W implementacji można korzystać tylko z elementarnych konstrukcji Python'a, takich jak: funkcje, instrukcje warunkowe, pętle.
2. Nie wolno korzystać z wbudowanych algorytmów sortowania.
3. 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

Napis nazywamy wielokrotnym, jeżeli powstał przez  $n$ -krotne ( $n > 1$ ) powtórzenie innego napisu o długości co najmniej 1. Przykłady napisów wielokrotnych: *ABCABCABC*, *AAAA*, *ABAABA*. Dana jest tablica  $T[N]$  zawierająca napisy. Proszę napisać funkcję `multi(T)`, która zwraca długość najdłuższego napisu wielokrotnego występującego w tablicy  $T$  lub wartość 0, jeżeli takiego napisu nie ma w tablicy.