

# ProgramowanieProceduralne

[Strona główna](#) / [Moje kursy](#) / [PP](#) / [LAB\\_13](#) / [IS\\_L7](#)

## IS\_L7

W PROGRAMACH MOŻNA KORZYSTAĆ Z WCZEŚNIEJ NAPISANYCH FUNKCJI/PROCEDUR DOŁĄCZAJĄCYCH ELEMENTY DO LISTY NA POCZĄTEK/KONIEC, ORAZ WYPISUJĄCYCH LISTĘ I ZWALNIAJĄCYCH PAMIĘĆ

Testowanie programu polega na wywołaniu funkcji/procedury dla przykładowych danych i wypisaniu na ekran wszystkich list, których zawartość jest istotna z punktu widzenia treści zadania.

Przypominam, że poprawne rozwiązanie to rozwiązanie bez wycieków pamięci. Program sprawdzający czy pamięć została poprawnie zwolniona *valgrind*

```
gcc prog.c -out program
valgrind ./program
```

1. Dana jest struktura

```
struct tnode {
    char value;
    struct tnode *next;
};
```

( 2 ) Proszę napisać i przetestować **funkcję**, która odwraca listę jednokierunkową (w miejscu - nie tworzymy nowej listy, nowych elementów; element ostatni staje się elementem pierwszym, a pierwszy ostatnim) zmieniając powiązania między elementami . Funkcja zwraca wskaźnik do odwróconej listy.

```
Przed: head -> ['f']-> ['d'] -> ['u'] -> ['i'] -> NULL
Po:      head -> ['i'] -> ['u'] -> ['d'] -> ['f'] -> NULL
```

2. Dana jest struktura

```
struct tnode {
    char value;
    struct tnode *next;
};
```

( 2 ) Proszę napisać i przetestować **procedurę**, która dodaje elementy do listy rosnąco.

```
Przed: head -> ['a']-> ['j'] -> ['k'] -> ['z'] -> NULL
Dodajemy:  ['w']
Po:      head -> ['a']-> ['j'] -> ['k'] -> ['w'] -> ['z'] -> NULL
```

3. Dana jest struktura

```
struct tnode {
    char value;
    struct tnode *next;
};
```

( 3 ) Proszę napisać i przetestować **funkcję** usuwającą elementy z listy.

Argumentem funkcji ma być wartość, na podstawie której dokonamy wyboru elementów do usunięcia. Niezależnie o tego czy argumentem jest mała czy duża litera usuwamy zarówno małą jak i dużą. Czyli gdy wywołamy funkcję dla 'a' usuwamy wszystkie elementy listy zawierające zarówno 'a' jak i 'A' ; podobnie gdy argumentem będzie 'A' .

Funkcja ma zwracać listę usuniętych elementów.

```
Przed: head -> ['a']->['t']->['o']->['k']->[' ']->['a']->['m']->[' ']->['a']->['l']->['A']-> Null

Wywołane funkcji dla liter 'a'
Po:      head -> ['t']->['o']->['k']->[' ']->['m']->[' ']->['l']-> Null
         left -> ['a']->['a']->['a']->['A']-> Null
```

4. Dana jest struktura

```
struct tnode {
    char value;
    struct tnode *next;
};
```

( 3 ) Proszę napisać i przetestować **funkcję**, która łączy dwie posortowane rosnąco listy jednokierunkowe w jedną również posortowaną rosnąco . Funkcja ma zwracać wskaźnik do pierwszego elementu wygenerowanej listy - rozwiązanie nie powinno tworzyć nowych elementów.

```
Przed:
lista_1 -> ['a']-> ['k'] -> ['l'] -> ['z'] -> NULL
lista_2 -> ['a']-> ['b'] -> ['n'] -> ['l'] -> ['w'] -> NULL
Dodajemy:  ['w']
Po:
lista -> ['a']-> ['a'] -> ['b'] -> ['k'] -> ['l'] -> ['l'] -> ['n'] -> ['w'] -> ['z'] -> NULL
lista_1 -> NULL
lista_2 -> NULL
```

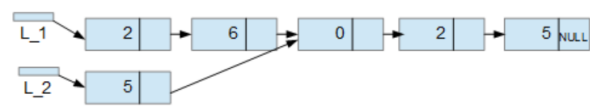
5. Dana jest struktura

```
struct tnode {
    int value;
    struct tnode *next;
};
```

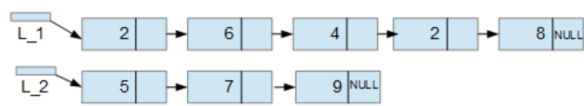
( 3 ) Proszę napisać program, który będzie wczytywał liczby naturalne < 20 (zakładamy, że podajemy liczby poprawne). W programie :

- należy utworzyć listę jednokierunkową **L\_1** złożoną z elementów typu **struct tnode**-- do których wpisujemy liczby parzyste, aż do momentu gdy wczytamy 0
- należy utworzyć listę **L\_2** złożoną z elementów typu **struct tnode** -- do których wpisujemy liczby nieparzyste, aż do momentu gdy wczytamy 0
- gdy wczytamy 0 element ten, oraz wszystkie kolejne mają być zapisane do obu list.
- gdy wczytamy wartość > 20 kończymy wczytywanie

Dla wczytanych liczb: 2, 5, 6, 0, 2, 5, 25



Dla wczytanych liczb: 2, 5, 6, 4, 2, 7, 9, 8, 25




Proszę napisać i przetestować **funkcję** stwierdzającą czy utworzone w programie listy **L\_1** oraz **L\_2** mają jakiś wspólny element (nie wartość typu **integer**, a element typu **struct tnode**).


## Status przesłanego zadania

Status przesłanego zadania	Przesłane do oceny
Stan oceniania	Nieocenione
Termin oddania	poniedziałek, 25 maja 2020, 14:25
Pozostały czas	Zadanie zostało złożone 8 min. przed terminem
Ostatnio modyfikowane	poniedziałek, 25 maja 2020, 14:17
Przesyłane pliki	<div><div>-  <a href="#">1</a> <a href="#">2</a> <a href="#">3</a> <a href="#">4.c</a></div><div>-  <a href="#">5.c</a></div></div> <div><div>25 maja 2020, 14:17</div><div>25 maja 2020, 14:17</div></div>

Komentarz do przesłanego zadania

- ▶

[Komentarze \(2\)](#)
- 

[Grażyna Krupińska](#) - nie, 31 maj 2020, 22:51  
zad4  
w trakcie łączenia list tworzone są nowe elementy (dodaj\_roznaco(&con, wsk->value);), a miało być tylko odpinanie i przypinanie
- 

[Grażyna Krupińska](#) - nie, 31 maj 2020, 22:51  
zad3  
tworzone są nowe elementy (dodaj\_na\_koniec(&new\_head, wsk->value);), a miało być tylko odpinanie i przypinanie

Dodaj komentarz ...

[Zapisz komentarz](#) | [Anuluj](#)

◀ [Obsługa Listy Jednokierunkowej](#)

Przejdź do...

[IS\\_L7](#) ▶



Platforma e-Learningowa obsługiwana jest przez:  
Centrum e-Learningu AGH oraz Uczelniane Centrum Informatyki AGH

[Podsumowanie zasad przechowywania danych](#)  
[Pobierz aplikację mobilną](#)