Luko Brasiūno

C KALBOS PROGRAMAVIMO STANDARTAS

Versija 1.0

TURINYS

TAISYKLĖS KODO LYGIAVIMUI

1. Funkcijų aprašai ir apibrėžimai rašomi prie pat kairiojo krašto.

```
3   void numberValidation(int *n);
4   void printSelectionMenu();
5   int checkMenu();
```

2. Funkcijose esantis kodas patraukiamas per vieną TAB‘ą nuo kairiojo krašto palyginus su funkcijos apibrėžimu.

```
54   void printSelectionMenu()
55  {
56      printf("Please select
57      printf("    1) Create
58      printf("    2) Print
```

3. Sąlygos sakiniuose ir cikluose esantis kodas irgi patraukiamas per vieną TAB‘ą nuo kairiojo krašto palyginus su šių eilučių padėtimi.

```
else
{
    printf("Wrong input!\n");
    while(!isspace(getchar()))
        ;
}
```

4. Skirtingą funkciją atliekantys kodo fragmentai atskiriami vienu tarpu (jeigu paskirtis nedaug skiriasi) ir keliais tarpais (jeigu paskirtis visiškai skiriasi).

```
// Assigning some information from the given node
number = current->value;
index = current->key;
current = current->next;

// Running a cycle until the last node in the list
while(current != NULL)
{
```

5. Funkcijų apibrėžimai atskiriami vienu tarpu.

```
 95      }
 96      return index;
 97  }
 98
 99  void deleteGivenListNode(struct
100  {
101      if(index >= amount)
102      {
```

6. Jeigu kodo eilutė ganėtinai ilga (reikia slinkti į kairę, kad perskaityti), eilutė dalinama į kelias dalis.

```
printf("This is the testing program for the linked list functions.\nPlease enter the "
       "following data: (1 7 4) or (1 2 5) in order for the testing program to work.\n");
struct Node* head = NULL;
```

TAISYKLĖS VARDŲ PARINKIMUI

1. Kintamieji rašomi angliškai.
```
int value;
int key;
```
2. Visi vienažodžiai ar keliažodžiai kintamieji vadinami pilnais, nesutrumpintais vardais (išskyrus 5-ame punkte nurodytus kintamuosius arba jeigu kintamasis susideda iš keturių ar daugiau žodžių).
```
int menuNumber;
```

3. Vieno žodžio kintamieji rašomi vien tik iš mažųjų raidžių.
```
int value;
int key;
```

4. Jeigu kintamasis iš dviejų ar daugiau žodžių, tai antras ir vėlesni žodžiai pradedami didžiosiomis raidėmis.
```
int menuNumber;
int findLowestValueInList
```

5. Laikini, pagalbiniai kintamieji vadinami *temp* trumpiniu prie kurio gali būti pridėtas papildomas pagalbinis žodis, padedantis identifikuoti kintamojo paskirtį.
```
int tempNumber;
```

6. Funkcijų pavadinimams galioja tos pačios, aukščiau nurodytos, taisyklės.
```
void printList(struct Node* head)
{
```

TAISYKLĖS KOMENTARAMS

1. Komentarai rašomi virš kodo fragmento ar eilutės.

```
// Prints the user options on screen
printSelectionMenu();
// Gets the user selection(input) from screen
int selection = checkMenu();
```

2. Komentarai rašomi angliškai.

```
// A function which prints out the options on screen
void printSelectionMenu()
{
```

3. Komentaro sakinio pradžios žodis rašomas iš didžiosios raidės.

```
// Data variable
int value;
// Node index variable
int key;
// Pointer to the next node
struct Node *next;
```

4. Jeigu komentaras ganėtinai ilgas – skaidomas į kelias dalis (eilutes).

```
// Assigning the first node data to a new temporary
//  Node* variable without altering the first node data
temp = *head;
```

5. Pačiame failo viršuje parašoma programos paskirtis.

```
1   // A file containing all of the functions for manipulating linked lists
2
3   #include <stdio.h>
4   #include <stdlib.h>
```

6. Nekomentuojamos akivaizdžią paskirtį atliekančios kodo eilutės ar fragmentai.

```
}
printf("Printing out list items:\n");
```

TAISYKLĖS KODUI

1. Jeigu cikle arba sąlygos sakinyje užtenka parašyti vieną kodo eilutę, riestiniai skliaustai nededami.

```
if(current == head)
    head = head->next;
if(current->key == index)
    previous->next = current->next;
```

2. Jeigu cikle ar sąlygos sakinyje reikia parašyti kelias kodo eilutes, tai riestiniai skliaustai dedami naujose eilutėse (pradžia ir pabaiga).

```
while(current != NULL)
{
    if(number > current->value)
    {
        number = current->value;
        index = current->key;
    }
}
```

3. Visos kode esančios operacijos (palyginimai, aritmetiniai veiksmai, prilyginimai ir t.t.) atskiriamos tarpu prieš ir po jomis.

```
while(current != NULL)
number = current->value;
index = current->key;
```

4. Jeigu sąlygos sakinyje tikimasi gauti nulinę (0) reikšmę, prieš tikrinamą kintamąjį rašomas ! simbolis.

```
if(!sizeValue)
```