

Студент: Пантюхин А.Е., группа: ДТ-460а

Лабораторная работа №2

Задание 1: Найти среднее арифметическое элементов списка

Задание 2: Определить, входит ли список L1 в L2, вернуть адрес начала вхождения

Тестовые данные:

Задание 1:

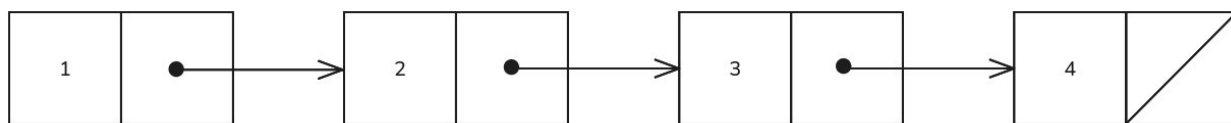
- 1) 1 2 3 4 5 6
- 2) -8 55 84 35 0 0 0
- 3) 1

Задание 2:

- 1) L1: 1 2 3, L2: 1 2 2 3
- 2) L1: 1, L2: 5 5 2 1 3
- 3) L1: 88 44 6 8 9 L2: 88 44 6 8 9

Графическая интерпретация списков:

head



Определения двух функций с комментариями:

//Вычисление среднего арифметического

double get_average(List* head)

{ //если списка на входе нет, выдаст 0

int elements = 0; //счётчик элементов

double sum = 0.0f; //сумма и результат(переиспользуется)

while (head)

{

sum += head->value; //+значение

head = head->next; //переход дальше

elements++; //+количество элементов

}

if (elements) sum /= elements; //на 0 делить нельзя!

return sum;

}

//поиск вхождения

//needle - «иголка», что ищем, haystack - «стог сена», в котором ищем.

List* lstlst(List* needle, List* haystack)

{

List* out = 0; //адрес вхождения

//некорректные данные - вернём 0 (вхождения нет)

if (!needle || !haystack) return out;

while (haystack && !out)

{ //пока есть где искать, и ещё не нашли

//если значения равны, то:

if (haystack->value == needle->value)

{

//снова идём вперёд

List *hptr = haystack, *nptr = needle;

//val_diff станет ненулевым, если значения отличаются

long long int val_diff = 0;

while (hptr && nptr && !val_diff)

{ //пока не отличаются...

val_diff = hptr->value - nptr->value;

hptr = hptr->next;

nptr = nptr->next;

}

//Если отличий нет, а список-вхождение уже кончился - значит, найдено

if (!val_diff && !nptr) out = haystack;

}

//движение дальше по списку в котором ищем вхождение

haystack = haystack->next;

}

//возвращаем адрес вхождения, если найден, иначе 0.

return out;

}