**Практическое задание 6**

**Тема.** Рекурсивные алгоритмы и их реализация.

**Цель.** Получить знания и практические навыки по разработке и реализации рекурсивных процессов.

**Задание.**

Разработать и протестировать рекурсивные функции в соответствии с задачами варианта. Составить отчет.

Требования к выполнению первой задачи варианта:

* приведите итерационный алгоритм решения задачи
* реализуйте алгоритм в виде функции и отладьте его
* определите теоретическую сложность алгоритма
* опишите рекуррентную зависимость в решении задачи
* реализуйте и отладьте рекурсивную функцию решения задачи
* определите глубину рекурсии, изменяя исходные данные
* определите сложность рекурсивного алгоритма, используя метод подстановки и дерево рекурсии
* приведите для одного из значений схему рекурсивных вызовов
* разработайте программу, демонстрирующую выполнение обеих функций, и покажите результаты тестирования.

Требования к выполнению второй задачи варианта:

* рекурсивную функцию для обработки списковой структуры согласно варианту. Информационная часть узла – простого типа – целого;
* определите глубину рекурсии
* определите теоретическую сложность алгоритма
* разработайте программу, демонстрирующую работу функций и покажите результаты тестов.

Таблица 1. Варианты заданий

|  |  |
| --- | --- |
| Номер | Задачи |
| 1 | 1. Найти наибольший общий делитель двух целых чисел 2. Создание и вывод линейного однонаправленного списка из n элементов |
| 2 | 1. Найти n-ое число Фибоначчи. 2. В однонаправленном списке из n элементов найти элемент с заданным значением и вернуть на него указатель. |
| 3 | 1. Определить делится ли число на каждую из своих цифр. 2. Не используя связный стек проверить баланс скобок в арифметическом выражении, которое передано как строка. |
| 4 | 1. Определить является ли текст – палиндромом. 2. Удалить из связанного однонаправленного списка все элементы, равные заданному. |
| 5 | 1. Дан массив из n элементов вещественного типа. Вычислить среднее значение всех элементов массива. 2. Создание связанного стека из n элементов. |
| 6 | 1. Сколько квадратов можно отрезать от прямоугольника со сторонами ***a*** и ***в.*** 2. Удаление связанного стека. |
| 7 | 1. Найти максимальный элемент в массиве из n элементов. 2. Создание очереди на однонаправленном списке. |
| 8 | 1. Перевести число из 10-системы счисления в систему с основанием В(1<В≤10) 2. Удаление очереди, реализованной на однонаправленном списке |
| 9 | 1. Бинарный поиск элемента в массиве 2. Создание двунаправленного списка. |
| 10 | 1. Вычислить значение цифрового корня для некоторого целого числа N. 2. Найти в двунаправленном списке количество четных элементов. |
| 11 | 1. Вычислить x1(x2+x3)(x4+x5+x6)....(x46+x47+...+x55). 2. Удаление двунаправленного списка |
| 12 | 1. Сортировка массива по возрастанию 2. Создать новый однонаправленный список из исходного однонаправленного списка, записав его элементы наоборот. |
| 13 | 1. Дана последовательность из N чисел Х1,Х2,....,ХN. Вычислить значение выражения: Хn(Хn+Xn-1)(Хn+Xn-1+Xn-2)(Хn+Xn-1+Xn-2+Xn-3)... (Хn+Xn-1+Xn-2+...+X1). Массив не использовать. 2. Удалить из однонаправленного списка нули. |
| 14 | 1. Дана строка. Выполнить переворот строки (записать наоборот) на ее же месте в памяти. 2. Определить количество вхождений: положительных, отрицательных, нулевых значений в линейном списке. |
| 15 | 1. Ханойская башня. 2. Удалить однонаправленный список. |

Вопросы по теме

1. Определение рекурсивной функции
2. Шаг рекурсии
3. Глубина рекурсии
4. Условие завершения рекурсии
5. Виды рекурсии
   1. - линейная -
   2. - каскадная –
6. Прямая и косвенная рекурсия
7. Организация стека рекурсивных вызовов

**Примеры реализации рекурсивных алгоритмов**

#include "stdafx.h"

#include "iostream.h"

// Задача 1.Дана последовательность целых чисел, заканчивающаяся нулем

// вывести сначала положительные, а затем отрицательные значения

void rec1();

Рекуррентная зависимость

//Задача 2. Вычислить xn. При x=0 и n<0 результат INFINITY

Такое определение алгоритма говорит об его рекурсивной природе

int rec2(int x, int n);

int main()

{

rec1();

cout<<rec2(2,3);

double rez=rec2(0, -3);

if (rez == INFINITY)

std::cout << "zero divide";

else

std::cout << rez;

return 0;

}

void rec1()

{int n;

cin>>n;

if (n==0)

return;

else

if(n>0)

{

cout<<n;

rec11();

}

else

{rec11();

cout<<n;

}

}

double rec2(int x, int n)

{

if (n==0)

{

return 1;

}

If (n>0)

// step recursii rec2(x,n)=x\*rec2(x,n-1)

return x\*rec2(x,n-1);

if(n<0){

return 1/rec2(x,abs(n));

}