Федеральное государственное автономное  
образовательное учреждение высшего образования  
"Национальный исследовательский университет   
"Высшая школа экономики"

*Факультет социально-экономических и компьютерных наук*

Косвинцев Степан Константинович

**ЛАБОРАТОРНАЯ РАБОТА № 2**

студента образовательной программы «Разработка информационных систем для бизнеса» по направлению подготовки *09.03.04 Программная инженерия*

Руководитель  
Дацун Н.Д.

Пермь, 2024 год

Оглавление

[1 Общая постановка задачи 3](#_Toc24542)

[2 Постановка задачи для варианта 11 3](#_Toc25355)

[3 Анализ задачи 3](#_Toc18133)

[4 Алгоритм 3](#_Toc23091)

[5 Программа 12](#_Toc4059)

[6 Результаты работы программы 12](#_Toc23501)

[ПРИЛОЖЕНИЕ А Текст программы 13](#_Toc15523)

Задача 1

Требуется построить рекурсивную формулу и вычислить функции для заданных значений (см. Таблицу 1).

Таблица 1.Функции и аргументы

|  |  |
| --- | --- |
| Функция | Аргументы |
| Add(x, y) = x + y | *x*= 4, *y*= 2 |
| *Mult*(*x*,*y*)  = *x* × *y* | *x*= 4, *y*= 2 |
| *Power*(*x*,*y*)  = *x**y* | *x*= 4, *y*= 2 |

Add

Рекурсивная формула

Add(x,y)=x+1+1…+1+1=(x+1+1…+1)+1=Add(x,y-1)+1=Inc(Add(x,y-1))

Add(x,0)=x

Add(x,y)=Inc(Add(x, y-1))

Add(4,2)

Результат равен 6. Дерево вызовов представленно на рисунке 1.

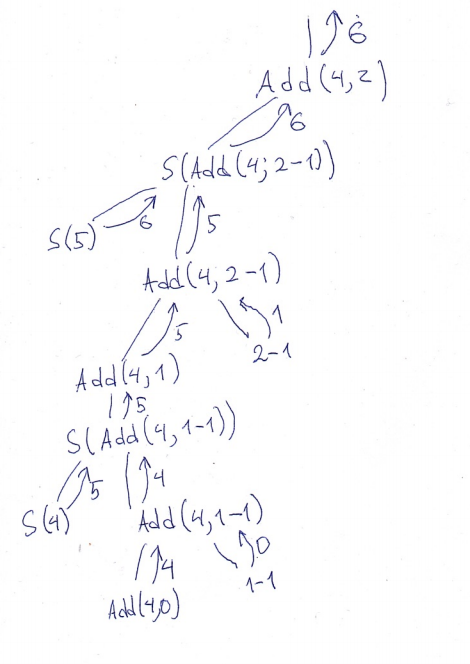


Рисунок 1. Вычисление Add(4,2)

Программа

#include <iostream>

int AddIter(int x, int y) {

int s = x;

for (int i = 0; i < y; i++) {

s++;

}

return s;

}

int AddReq(int x, int y) {

if (y == 0) {

return x;

}

return AddReq(x + 1, y - 1);

}

int main()

{

std::cout << AddIter(4,2)<<' '<<AddReq(4,2)<<'\n';

}

Пошаговое выполнение

Mult

Формулы

Mult(x,y)=x+x+x+..x=(x+x+x+…)+x=Add(Mult(x,y-1),x)

Mult(x,0)=0

Mult(x,y)=Add(Mult(x,y-1),x)

Mult(4,2)

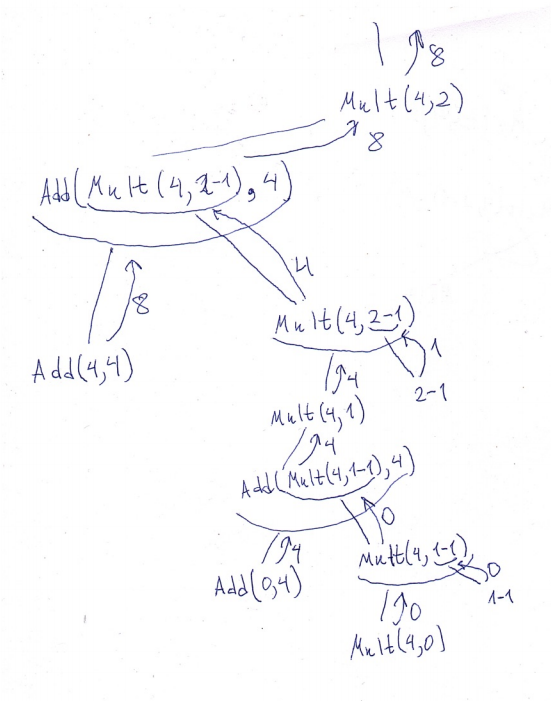


Рисунок 2. Дерево рекурсии

Код

#include <iostream>

int MultIter(int x, int y){

int res=0;

for(int i=0; i<y; i++){

res=res+x;

}

return res;

}

int MultReq(int x, int y){

if (y==0) return 0;

return MultReq(x,y-1)+x;

}

int main()

{

std::cout<<MultReq(4,2)<<' '<<MultIter(4,2)<<'\n';

return 0;

}

Трассировка

Pow

Формула

Pow(x,y)=x\*x\*x\*…\*x=(x\*x\*x\*…)\*x=Pow(x,y-1)\*x

Pow(x,0)=1

Pow(x,y)=Pow(x,y-1)\*x

Pow(4,2)

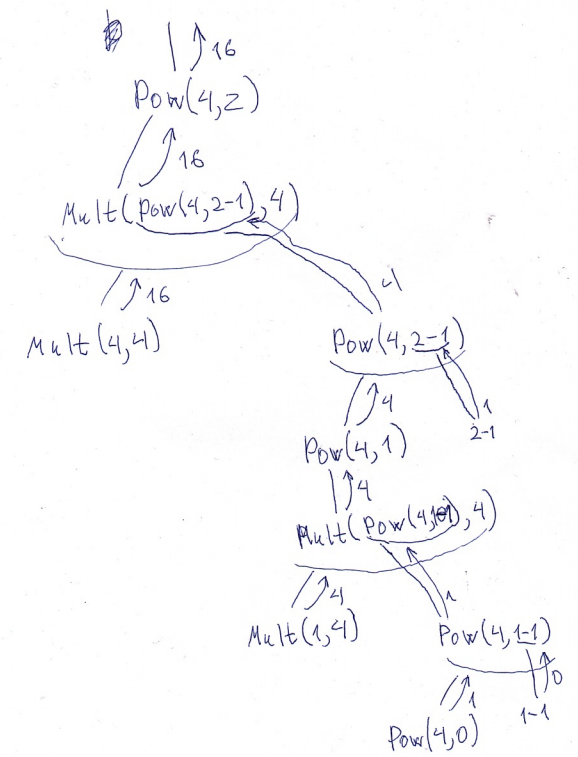


Рисунок 3

Код

#include <iostream>

int PowIter(int x, int y){

int res=1;

for(int i=0; i<y; i++){

res=res\*x;

}

return res;

}

int PowReq(int x, int y){

if (y==0) return 1;

return PowReq(x,y-1)\*x;

}

int main()

{

std::cout<<PowReq(4,2)<<' '<<PowIter(4,2)<<'\n';

return 0;

}

Трассировка

Задача 2

Fib(n) = 1, если n = 1 или n = 2,

иначе Fib(n) = Fib(n – 2) + Fib(n – 1).

Код

#include <iostream>

#include <vector>

#include <tuple>

using namespace std;

int FibReq(int n) {

if (n == 1 || n == 2) {

return 1;

}

return FibReq(n - 2) + FibReq(n - 1);

}

int FibIter(int n) {

vector<int> nums { 1,1 };

while (n > nums.size()) {

int a = nums[nums.size() - 1];

int b = nums[nums.size() - 2];

int nxt = a + b;

nums.push\_back(nxt);

}

return nums[n - 1];

}

Дерево вызовов Fib(6)

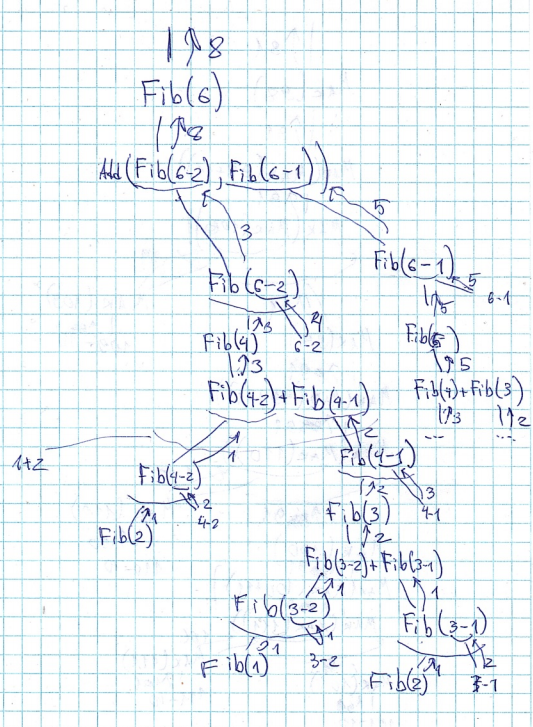


Рисунок 4

Трассировка

Задача 3

Маккарти(n) = n –10, если n>100,

иначе Маккарти(n) = Маккарти(Маккарти(n+11)).

**Код**

#include <iostream>

#include <vector>

#include <tuple>

using namespace std;

int mac(int n) {

if (n > 100) {

return n - 10;

}

return mac(mac(n + 11));

}

Дерево вызовов Mac(98)

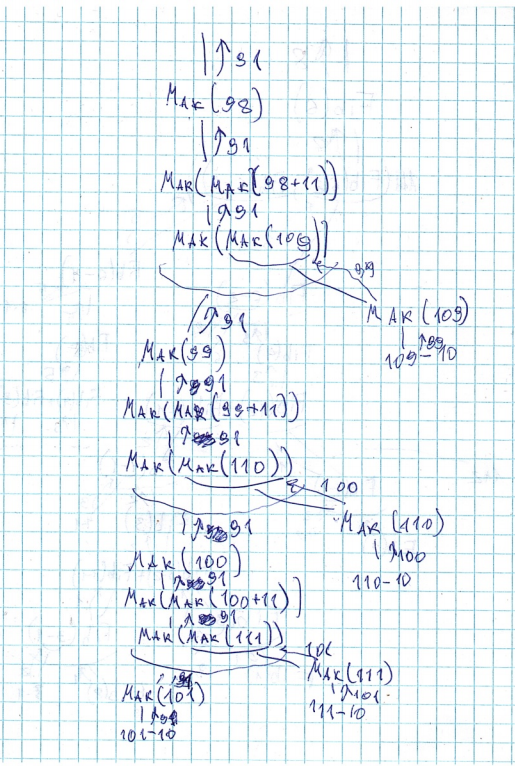


Рисунок 5

Вызов с другими значениями

Мои Эксперементы

|  |  |
| --- | --- |
| Значение N | Результат |
| 1 | 91 |
| 100 | 91 |
| 101 | 91 |
| 102 | 92 |
| 120 | 110 |

При любом N<101 Mak(n)=91

При n>100 Mak(n) = n-10

Задача 4

Аккерман(m, n) = n+1, если m = 0 или

иначе Аккерман(m, n) = Аккерман(m–1, 1), если n = 0 или

иначе Аккерман(m, n) = Аккерман(m–1, Аккерман(m, n–1))

Код

#include <iostream>

#include <vector>

#include <tuple>

using namespace std;

int acc(int m, int n) {

if (m == 0) {

return n + 1;

}

if (n == 0) {

return acc(m - 1, 1);

}

return acc(m - 1, acc(m, n - 1));

}

Дерево вызовов

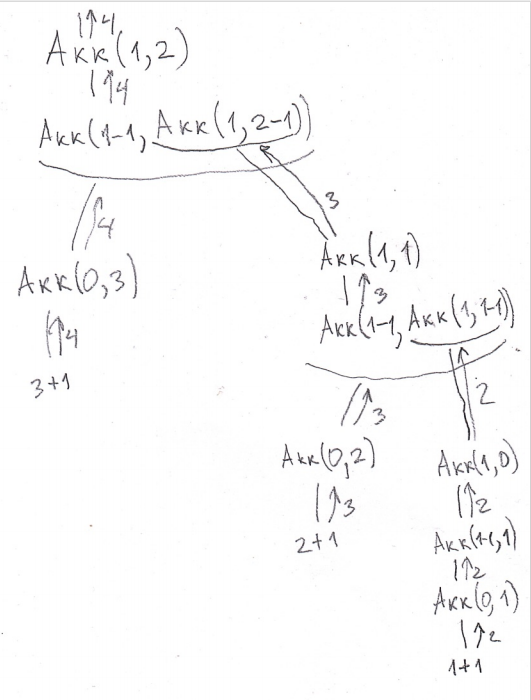


Рисунок 6

Трассировка

Задание 4

Раскраска

Код

#include <iostream>

#include <vector>

#include <tuple>

using namespace std;

vector<vector<int>> input\_field(int bordercolor, int fieldcolor) {

int m, n;

cout << "Введите количество строк\n";

cin >> m;

cout << "Введите количество столбцов\n";

cin >> n;

cout << "Введите фигуру по строкам, числа - номера цветов, "

<< bordercolor << " - цвет границы, "

<< fieldcolor << " - цвет закраски"<<'\n';

vector<vector<int>> res{};

for (int i = 0;i < m;i++) {

res.push\_back(vector<int>(n,0));

for (int j = 0;j < n;j++) {

cin >> res[i][j];

}

}

return res;

}

tuple<int,int> input\_start\_point() {

cout << "введите координаты стартовой точки закраски в формате x y\n";

int x, y;

cin >> x;

cin >> y;

return tuple<int, int>{ x - 1, y - 1 };

}

void fill\_field(int x, int y, vector<vector<int>>& field, int bordercolor, int fieldcolor, int fillcolor) {

if (x<0 || y<0 || x>=field.size() || y>=field[0].size())

return;

if (field[x][y] == fieldcolor) {

field[x][y] = fillcolor;

fill\_field(x + 1, y, field, bordercolor, fieldcolor, fillcolor);

fill\_field(x - 1, y, field, bordercolor, fieldcolor, fillcolor);

fill\_field(x, y + 1, field, bordercolor, fieldcolor, fillcolor);

fill\_field(x, y - 1, field, bordercolor, fieldcolor, fillcolor);

}

}

void fill\_process() {

int fillcolor = 1, bordercolor = 2, fieldcolor = 0;

vector < vector<int >> field = input\_field(bordercolor, fieldcolor);

int x, y;

tie(x, y) = input\_start\_point();

fill\_field(x, y, field, bordercolor, fieldcolor, fillcolor);

for (vector<int> v : field) {

for (int i : v) {

cout << i << ' ';

}

cout << '\n';

}

}

int main()

{

setlocale(LC\_ALL, "");

fill\_process();

}