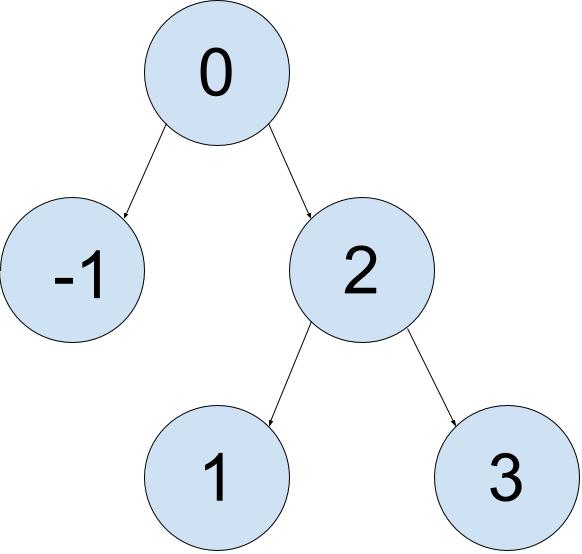
**№001 (4 балла)**

****

#include <iostream>

using namespace std;

struct Tree

{

int key;

Tree\* left;

Tree\* right;

void Insert(int);

}root;

void Output(Tree\* p)

{

cout << p->key << endl;

if (p->left) Output(p->left);

if (p->right) Output(p->right);

}

void main()

{

Output(&root);

}

Дано дерево с 5 вершинами. Сколько раз будет вызвана функция Output()?

Данная функция будет вызвана 5 раз: один раз из главной функции, два раза из корня и два раза из вершины с ключом 2, так как только эти вершины имеют потомков.

**№002 (4 балла)**

#include <iostream>

using namespace std;

void f1(int i)

{

cout << i++ << endl;

f2(i);

}

void f2(int i)

{

cout << i++ << endl;

f1(i);

}

void main()

{

f1(1);

}

Выполнится ли данная программа?

Данная программа не выполнится, так как определение функции f2() находится после её первого вызова, и компилятор выдаст ошибку



**№003 (5 баллов)**

#include <iostream>

using namespace std;

float f(float x, float sum)

{

float j = 1 / x;

sum += j;

while (j) f(x\*x, sum);

return sum;

}

void main()

{

cout << f(2, 0);

}

Будет ли выведено что-либо в консоль?

Ничего выведено не будет, так как цикл while внутри функции, в которой j не равно нулю, будет бесконечно вызывать эту функцию. Для корректной работы программы его необходимо заменить на if.

 с while

 с if

**#004 (4 балла)**

#include <iostream>

using namespace std;

void f(int i)

{

cout << i << endl;

f(++i);

}

void main()

{

f(1);

}

Будет ли выполняться данная функция бесконечное число раз? Почему?

Данная функция не будет выполняться бесконечно, так как рано или поздно (примерно через 4.5 тысячи вызовов) стек переполнится, и работа программы будет завершена.



**№005 (3 балла)**

#include <iostream>

#include <ctime>

using namespace std;

int difficult\_function(int i)

{

return (pow(i, exp(i \* pow(i, 987))) - pow(i, exp(i \* pow(i, 987))) + i);

}

double Sum(int i, double sum)

{

sum += difficult\_function(i);

i++;

if (i < 4000) return Sum(i, sum);

else return sum;

}

void main()

{

double sum = 0;

int start\_time = clock();

sum = Sum(1, 0);

int end\_time = clock();

cout << sum << " " << end\_time - start\_time << "ms" << endl;

sum = 0;

start\_time = clock();

for (int i = 0; i < 4000; i++)

{

sum += difficult\_function(i);

}

end\_time = clock();

cout << sum << " " << end\_time - start\_time << "ms" << endl;

}

Какой алгоритм будет выполнен быстрее? Почему?

Быстрее будет выполнен итерационный алгоритм, так как вызов функции требует дополнительных затрат, в частности, на перемещение указателя по памяти, в то время как обыкновенный цикл выполняет одну и ту же операцию раз за разом



**№006 (4 балла)**

struct node

{

int key = 0;

node \*left, \*right;

};

void output(node \*ptr)

{

cout << "key: " << ptr.key << endl;

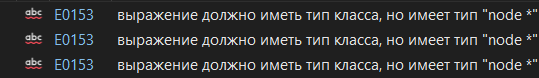
output(ptr.left);

output(ptr.right);

}

Выполнится ли данная функция? Почему?

Данная функция не выполнится, так как она принимает на вход указатель на тип node, а вызов осуществляется при помощи операции точка, неприменимого к указателю.



**№007 (5 баллов)**

#include <iostream>

using namespace std;

int f(int i)

{

float j = i;

return f(j);

}

float f(float i)

{

return i;

}

void main()

{

cout << f(1);

}

Что будет выведено в консоль?

В консоль ничего не будет выведено, так как функция, возвращающая float, определена после функции, вызываемой из главной функции, и отсутствует прототип, значит, первая функция будет вызывать сама себя, пока не переполнится стек



**№008 (3 балла)**

#include <iostream>

#include <ctime>

using namespace std;

int f(int i)

{

return i ? f(i - 3) : i;

}

void main()

{

cout << f(1024);

}

Что будет выведено в консоль?

В консоль ничего не будет выведено, так как для данного аргумента, передаваемого из главной функции, пропускается условие завершения рекурсии, и функция будет вызывать сама себя до тех пор, пока не заполнится стек



**№009 (4 балла)**

#include <iostream>

using namespace std;

struct Tree

{

int key;

Tree\* left;

Tree\* right;

}root = {0, 0, 0};

void Output(Tree \*&p)

{

cout << p->key << endl;

if (p->left) Output(p = p->left);

if (p->right) Output(p = p->right);

}

void main()

{

Tree\* ptr = &root;

Insert(ptr, 4);

Insert(ptr, -1);

Insert(ptr, 2);

Insert(ptr, 5);

Tree \*&p = ptr;

Output(p);

}

Что будет выведено в консоль?

Будет выведен только корень и его левый сын, так как при вызове функции для обхода левого поддерева указателю будет присвоено новое значение, и правое поддерево не будет выведено



**№010 (3 балла)**

void Output(Tree \*p)

{

if (p->left) Output(p->left);

cout << p->a;

if (p->right) Output(p->right);

}

Дано дерево (A(B(D,E(F)),C(G(I,J),H)))

В каком порядке будут выведены вершины при помощи данной функции?

Так как эта функция реализует обход ЛКП, то вершины будут выведены в следующем порядке: DBEFAIGJCH



**№011 (3 балла)**

#include <iostream>

using namespace std;

int f(int i)

{

if (i < 65) return f(i \* i);

else return i;

}

void main()

{

cout << f(2);

}

Что будет выведено в консоль?

В консоль будет выведено число 256, так как в функцию f будут передаваться следующие числа: 2, 4=22, 16=42, 256=162. Число 256 больше 65, а 16 меньше, поэтому функция вернёт именно 256



**№012 (3 балла)**

#include <iostream>

using namespace std;

int f(int i)

{

if (i) return i \* f(i - 1);

else return 1;

}

void main()

{

int n;

cin >> n;

cout << f(n);

}

Что выводит в консоль данная программа?

Данная программа выводит в консоль факториал введённого числа, так как факториал числа n определяется как n \* (n – 1)!, а 0!=1



**№013 (5 баллов)**

#include <iostream>

using namespace std;

int f(int n[], int sum, int i)

{

sum += \*(n + i);

if (i < 5) return f(n, sum, ++i);

else return sum;

}

void main()

{

int sum = 0;

int n[] = { 1,2,3,4,5 };

cout << f(n, sum, 0);

}

Что выведет данная программа?

Данная программа выведет мусорное значение, так как при i = 5 прибавление к sum неизвестного числа из памяти происходит до проверки окончания поданного на вход массива.



**№014 (3 балла)**

#include <iostream>

using namespace std;

int f(int n[], int sum, int i)

{

if (i < 5)

{

sum += \*(n + i);

return f(n, sum, i + 1.75);

}

else return sum;

}

void main()

{

int sum = 0;

int n[] = { 4,3,7,6,5 };

cout << f(n, sum, 0);

}

Что будет выведено в консоль?

Будет выведена сумма элементов переданного массива, несмотря на то, i увеличивается больше, чем на 1, так как перед передачей в функцию дробная часть отбрасывается



**№015 (6 баллов)**

void set(void\* root, int key, string str)

{

node\* p = new node;

(\*p).key = key;

(\*p).str = str;

(\*p).left = NULL;

(\*p).right = NULL;

if ((\*((node\*)root)).key > key) (\*((node\*)root)).left = p;

else if ((\*((node\*)root)).key < key) (\*((node\*)root)).right = p;

}

int f(void\* root, string str, int key)

{

if (key > (\*((node\*)root)).key)

{

if ((\*((node\*)root)).right) f((void\*)(\*((node\*)root)).right, str, key);

else set(root, key, str);

}

else if (key < (\*((node\*)root)).key)

{

if ((\*((node\*)root)).left) f((void\*)(\*((node\*)root)).left, str, key);

else set(root, key, str);

}

else

{

cout << "error\n";

exit(1);

}

}

Какую операцию реализует функция f?

Данная функция реализует операцию вставки значения в бинарное дерево поиска.