**20 билет**

/\*

предполагается, что есть структура

struct Tree

{

int data = 0;

Tree \*left = NULL, \*right = NULL;

};

\*/

//t - узел, с которого начинать изменение

void change(Tree \*t)

{

if (t == NULL)

return;

if (t->data % 2 == 0)

t->data \*= 2;

change\_(t->left);

change\_(t->right);

return;

}

**21 билет**

/\*

предполагается, что есть структура

struct Tree

{

int data = 0;

Tree \*left = NULL, \*right = NULL;

};

\*/

//t - всё дерево

void change(Tree \*t)

{

if (t == NULL)

return;

if (t->left == NULL && t->right == NULL)

{

if (t->data % 2 == 0)

{

t->left = new Tree;

t->left->data = t->data / 2;

}

return;

}

change(t->left);

change(t->right);

return;

}

**22 билет**

/\*

предполагается, что есть структура

struct Tree

{

int data = 0;

Tree \*left = NULL, \*right = NULL;

};

\*/

//t - всё дерево

void change(Tree \*t)

{

if (t == NULL)

return;

if (t->left == NULL && t->right == NULL)

{

if (t->data % 2 == 1)

{

t->right = new Tree;

t->right->data = t->data \* 2;

}

return;

}

change(t->left);

change(t->right);

return;

}

**23 билет**

/\*

предполагается, что есть структура

struct Tree

{

int data = 0;

Tree \*left = NULL, \*right = NULL;

};

\*/

//t - узел, с которого начинать подсчёт

int count(Tree \*t)

{

if (t == NULL)

return 0;

return

count(t->left) + count(t->right) +

((t->left != NULL) && (t->right != NULL))

;

}