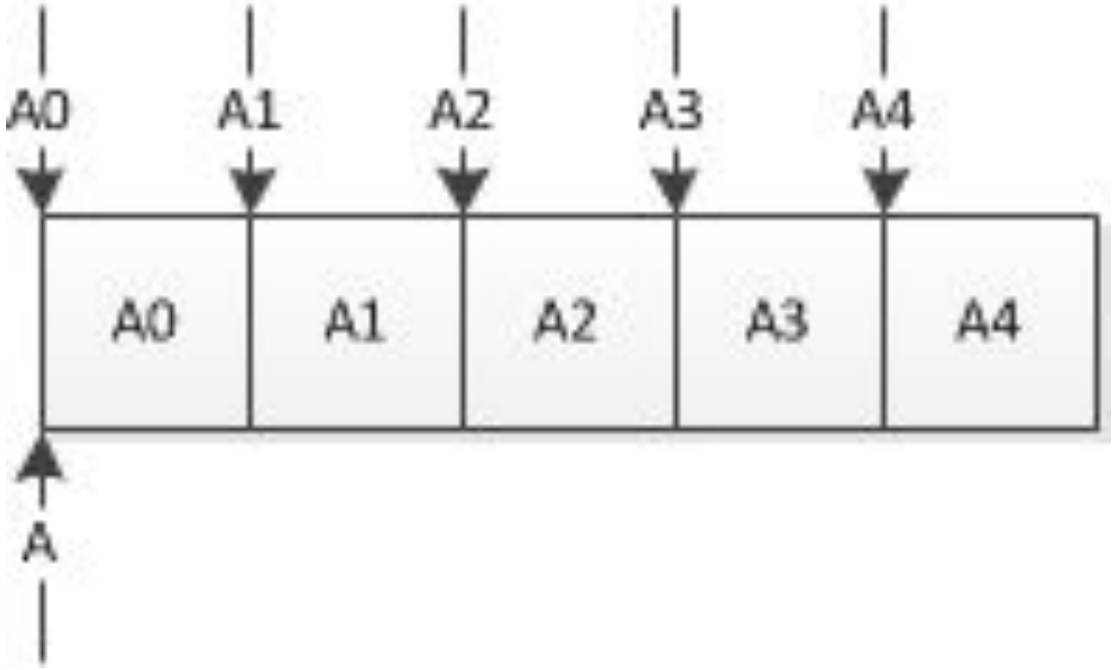
Односвязные списки

Массив

- Набор **последовательно расположенных в памяти однотипных** элементов

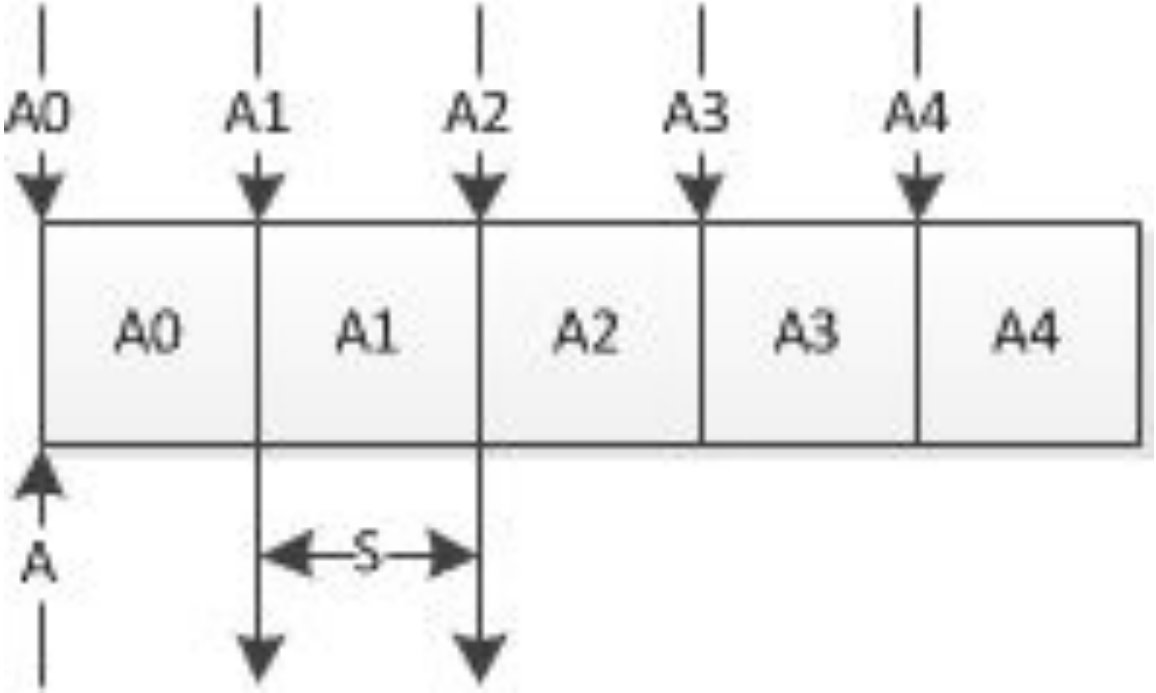
Свойства массива

- Быстрый доступ к первому элементу



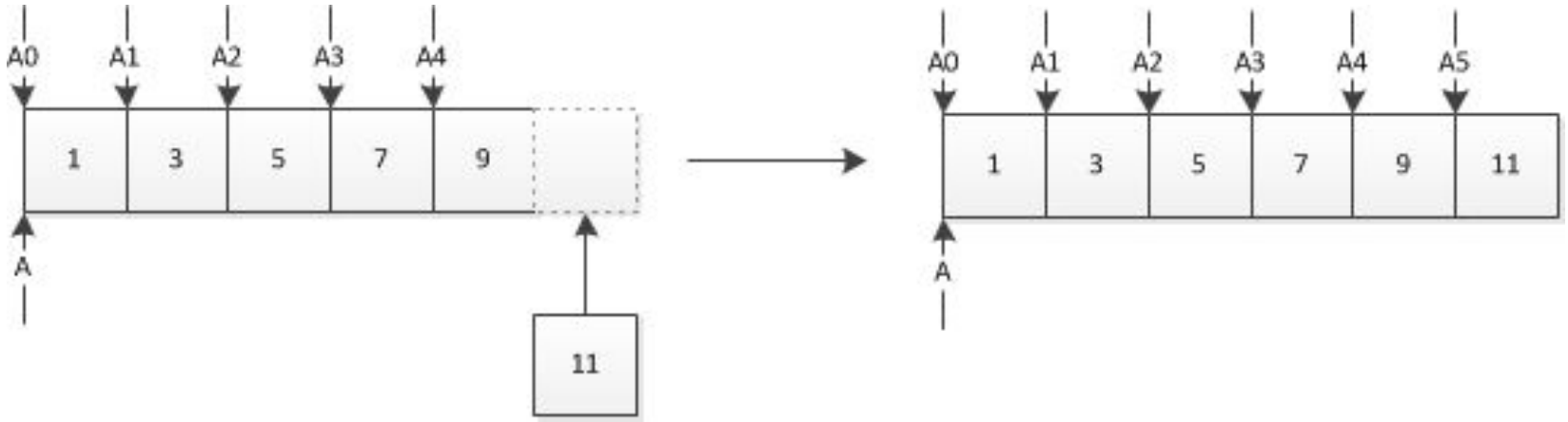
Свойства массива

- Быстрый доступ к элементу по индексу



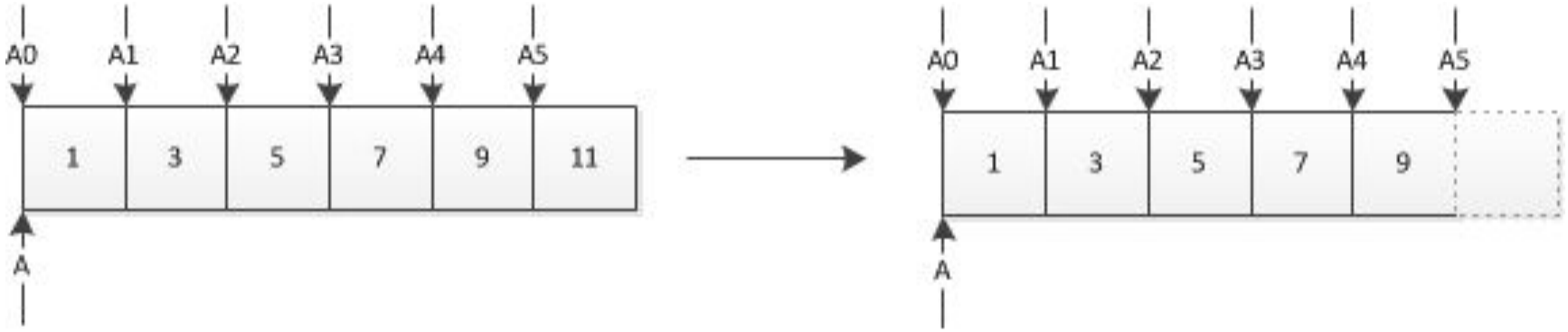
Свойства массива

- Быстрая вставка элемента в конец



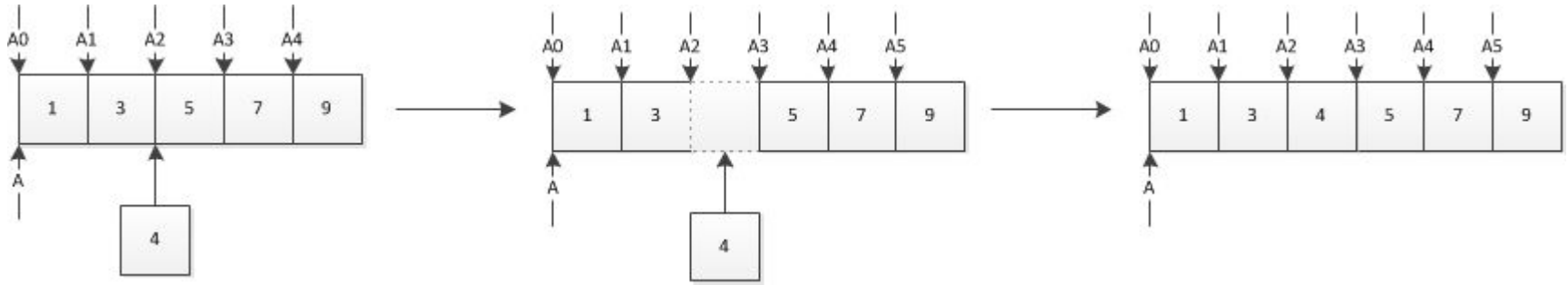
Свойства массива

- Быстрое удаление последнего элемента



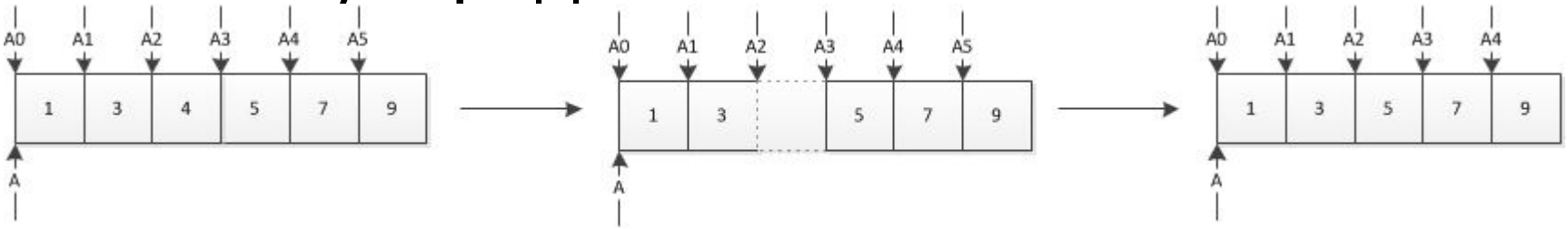
Свойства массива

- Медленная вставка в начало/середину



Свойства массива

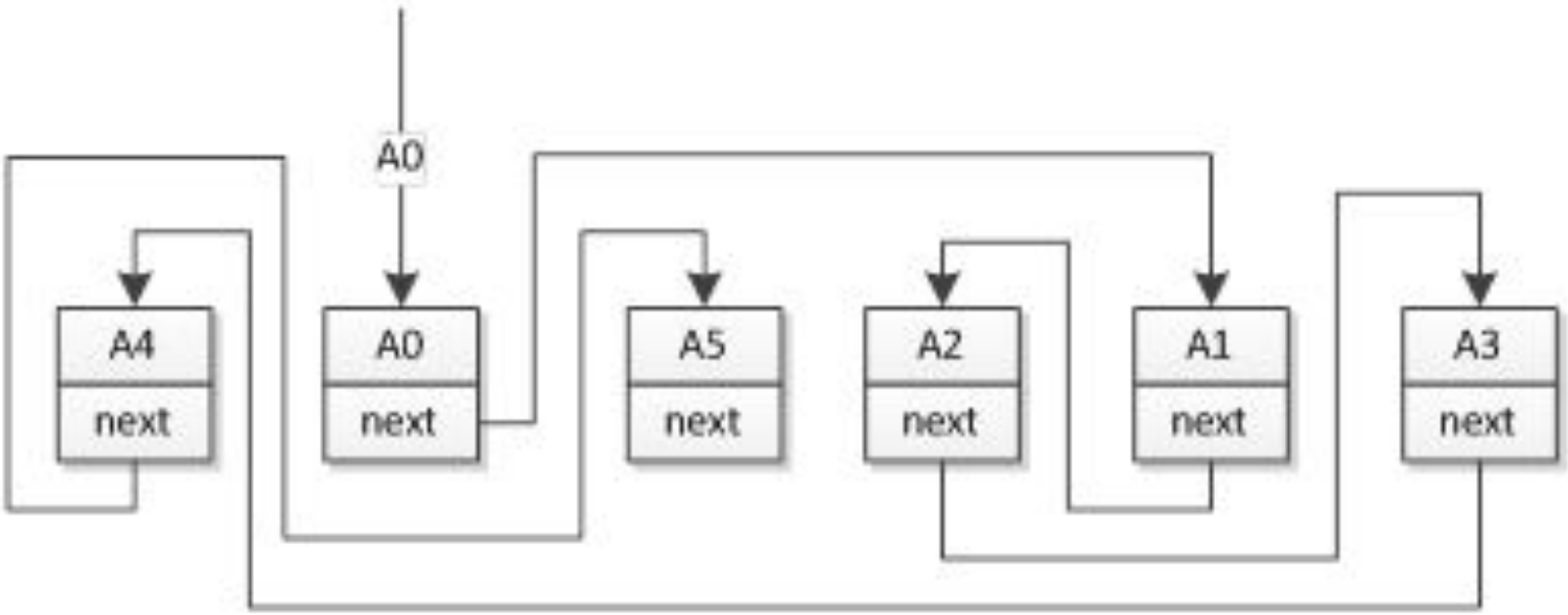
- Медленное удаление из начала/середины



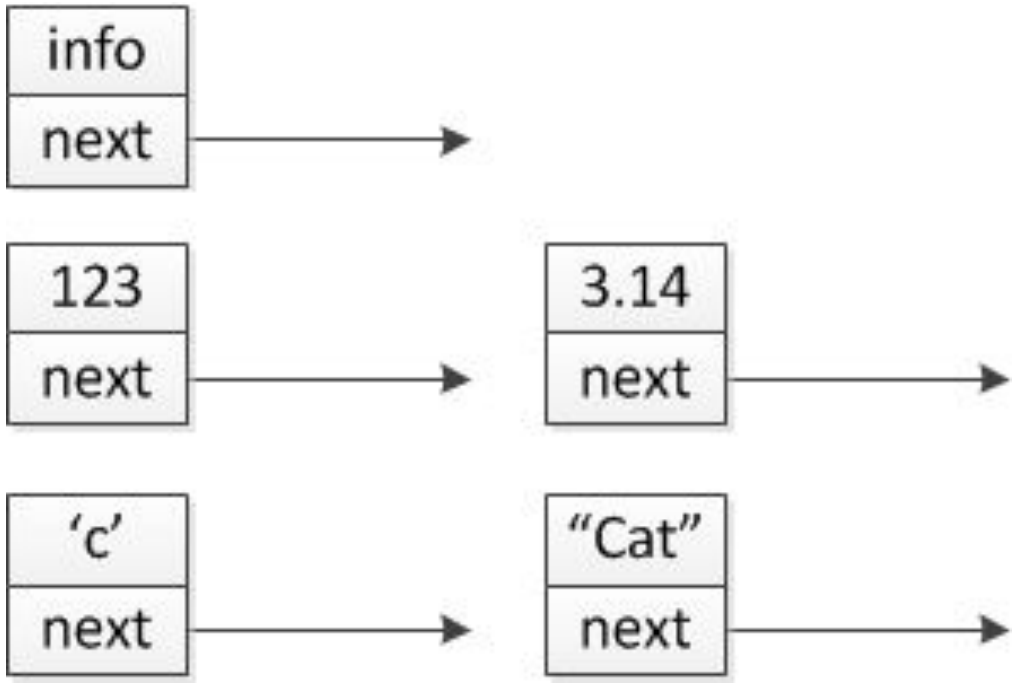
Резюме

- Быстрый доступ к элементу по индексу
- Медленное добавление в начало и середины
- Медленное удаление из начала и середины
- Трудности с выделением памяти и расширением

Идея списка



Узел списка и его содержимое



Узел списка (код, картинка)

```
struct node{
    int info;
    node* next;
};
```

Список руками

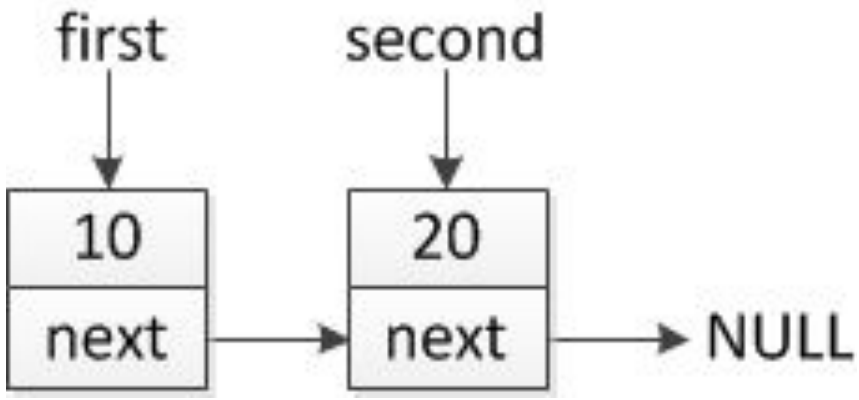
```
struct node{
    int info;
    node *next;
};
int main() {
    node *first, * second;
    first = new node;
    first->info = 10;
    second = new node;
    second->info = 20;
    first->next = second;
    second->next = NULL;
```

```
cout << first->info << endl;
cout << second->info << endl;
cout << first->next->info << endl;
cout << second->next->info << endl;
}
```

Список руками

```
struct node{
    int info;
    node *next;
};
int main() {
    node *first, * second;
    first = new node;
    first->info = 10;
    second = new node;
    second->info = 20;
    first->next = second;
    second->next = NULL;
```

```
cout << first->info << endl;
cout << second->info << endl;
cout << first->next->info << endl;
cout << second->next->info << endl;
}
```



Список руками 2

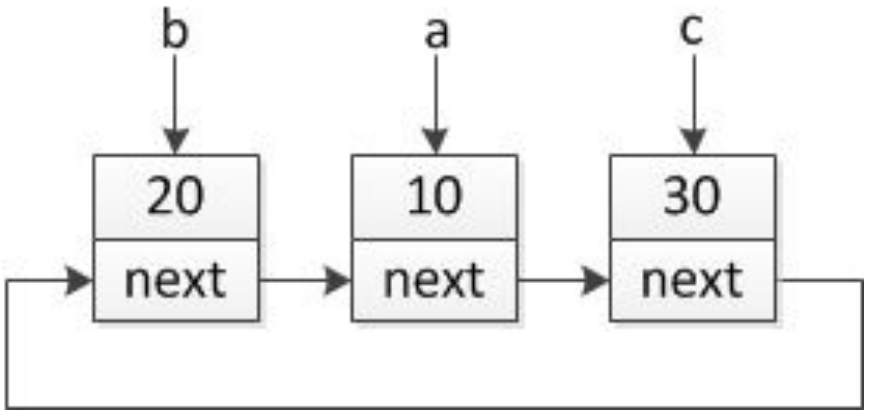
```
node *a, *b, *c;
a = new node;
a->info = 10;
b = new node;
b->info = 20;
c = new node;
c->info = 30;
a->next = c;
b->next = a;
c->next = b;
```

```
cout << a->info << endl;
cout << b->info << endl;
cout << c->info << endl;
cout << a->next->info << endl;
cout << a->next->next->info << endl;
cout << b->next->info << endl;
cout << b->next->next->info << endl;
cout << c->next->info << endl;
cout << c->next->next->info << endl;
```

Список руками 2

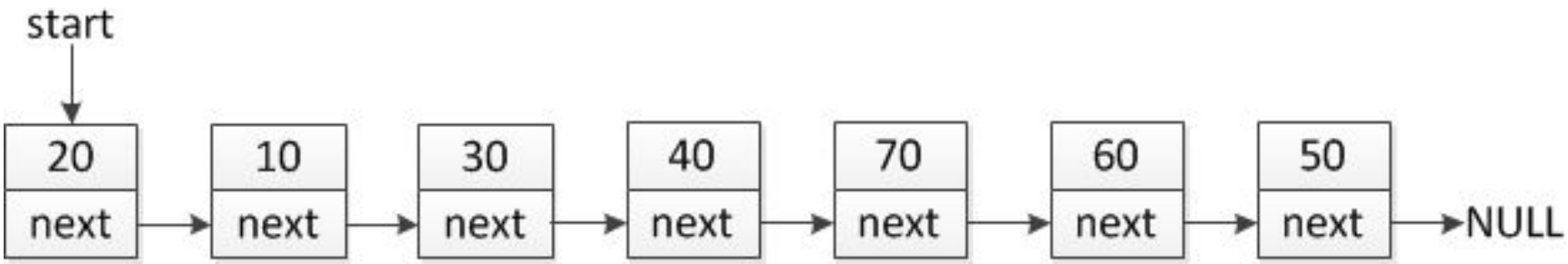
```
node *a, *b, *c;
a = new node;
a->info = 10;
b = new node;
b->info = 20;
c = new node;
c->info = 30;
a->next = c;
b->next = a;
c->next = b;
```

```
cout << a->next->info << endl;
cout << a->next->next->info << endl;
cout << b->next->info << endl;
cout << b->next->next->info << endl;
cout << c->next->info << endl;
cout << c->next->next->info << endl;
```

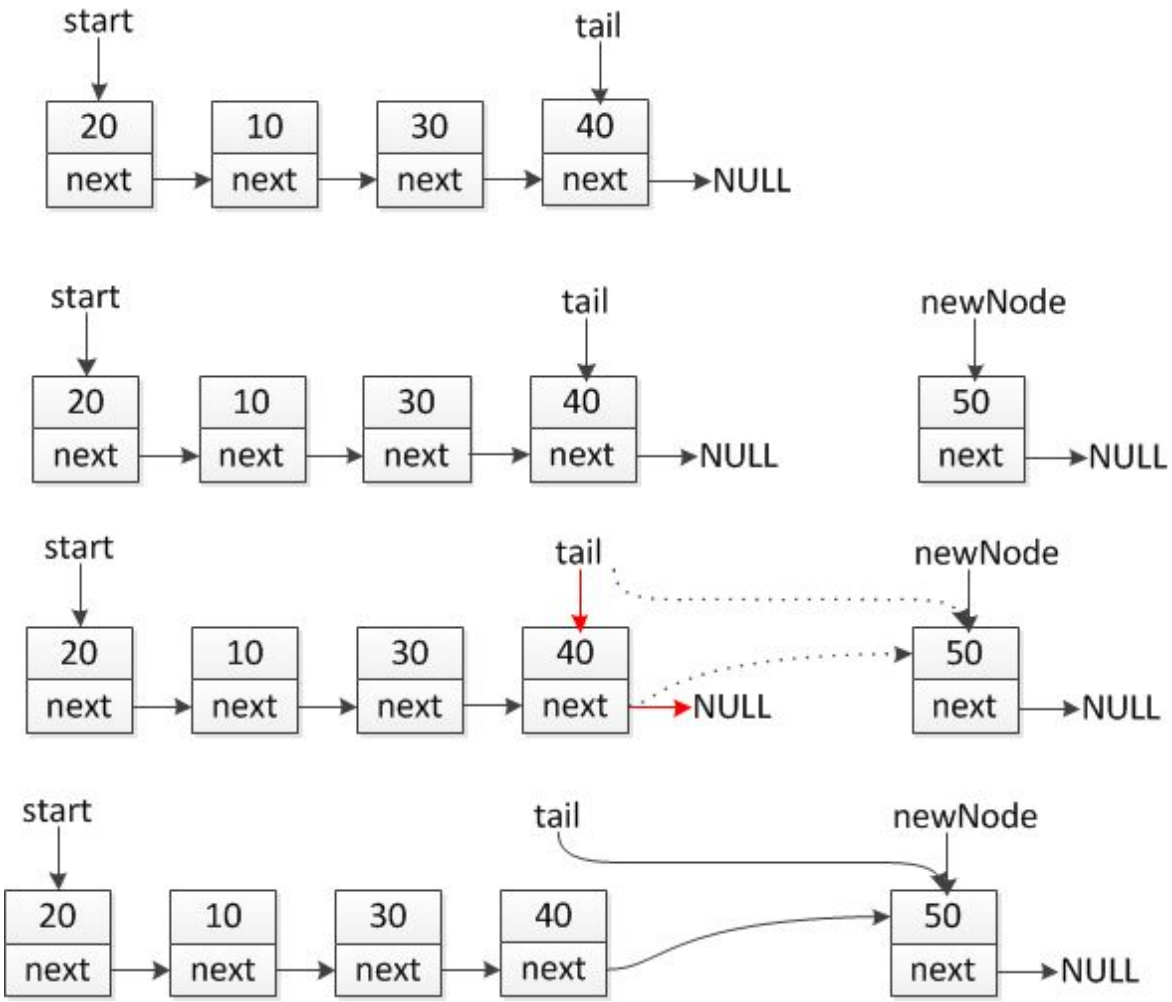


Вывод списка

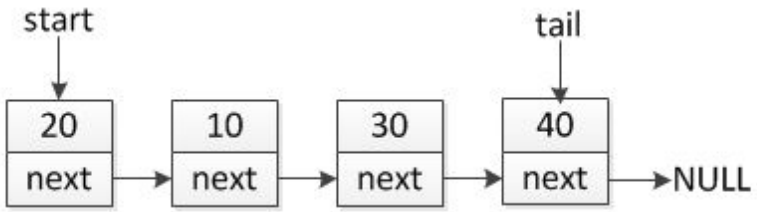
```
for(node* cur = start; cur!=NULL; cur = cur->next){
    cout << cur->info << endl;
}
```

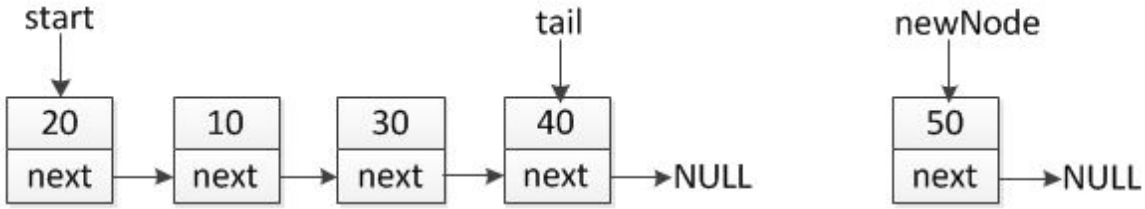


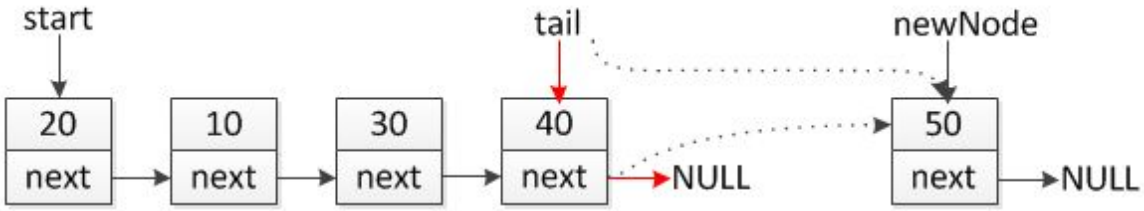
Дополнение списка

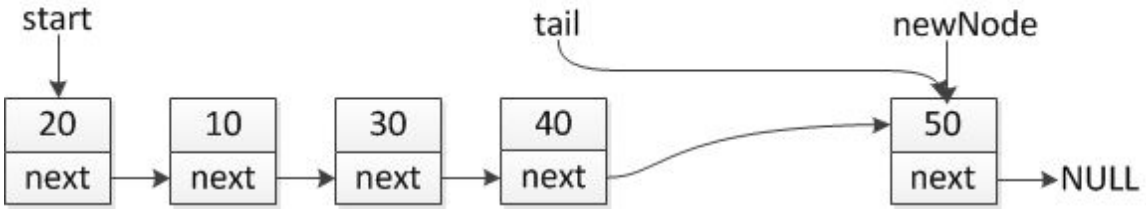


Дополнение списка



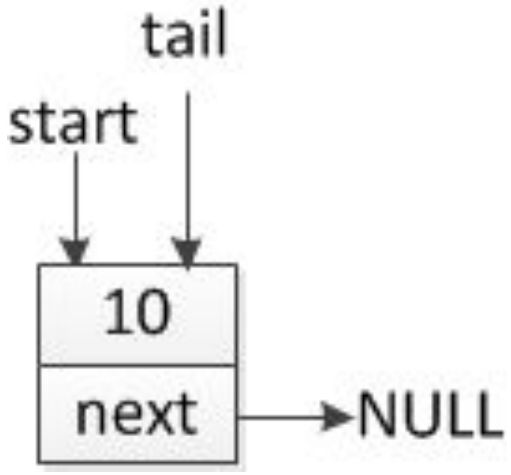







```
newNode = new node;
newNode->info = 50;
newNode->next = NULL;

tail->next = newNode;
tail=newNode;
```

Создание списка с нуля




```
node *newNode = new node;
newNode->info = 10;
newNode->next = NULL;
start = newNode;
tail = newNode;
```

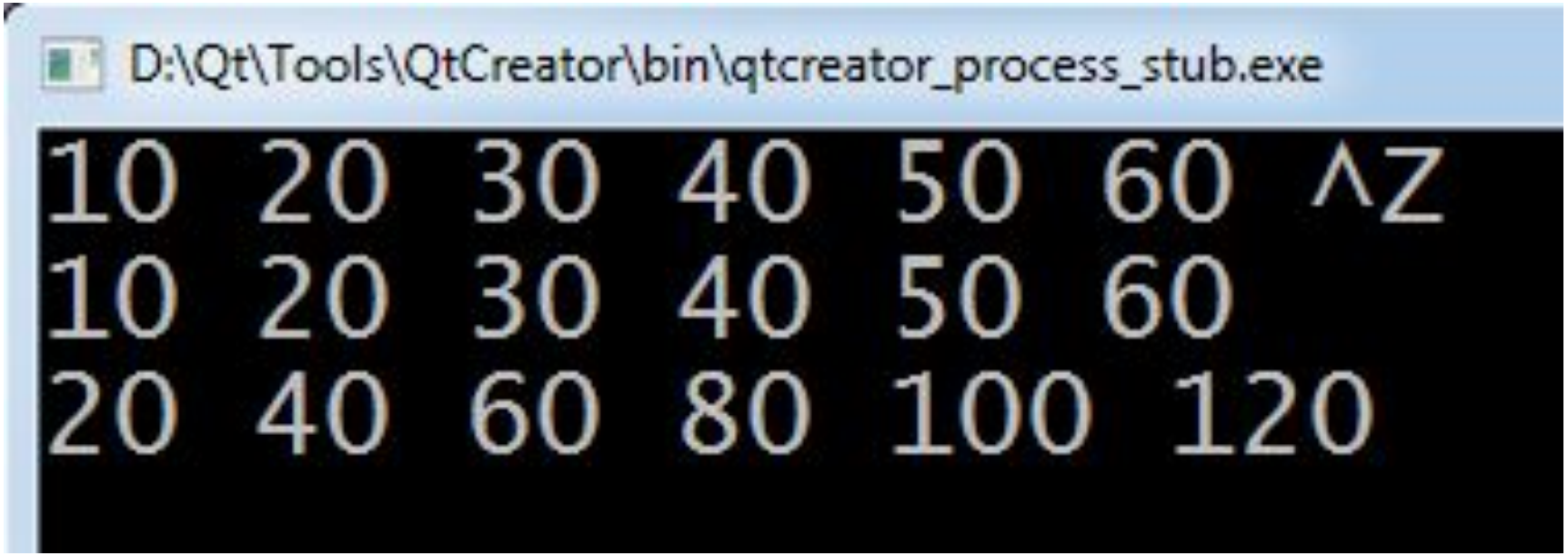
Ввод-обработка-вывод

```
node *head=NULL, *tail=NULL, *newNode;  int x;
while(cin>>x){
        newNode = new node;
        newNode->info = x;
        newNode->next = NULL;
        if(tail==NULL){
                head = newNode;
                tail = newNode;
        } else {
                tail->next = newNode;
                tail = newNode;
        }
}
```

Ввод-обработка-вывод

```
for(node* cur = head; cur!=NULL; cur=cur->next){
	cout << cur->info << " ";
}
cout << endl;

for(node* cur = head; cur!=NULL; cur=cur->next){
	cur->info *= 2;
}

for(node* cur = head; cur!=NULL; cur=cur->next){
	cout << cur->info << " ";
}
cout << endl;
```

Результат работы



Стек vs очередь

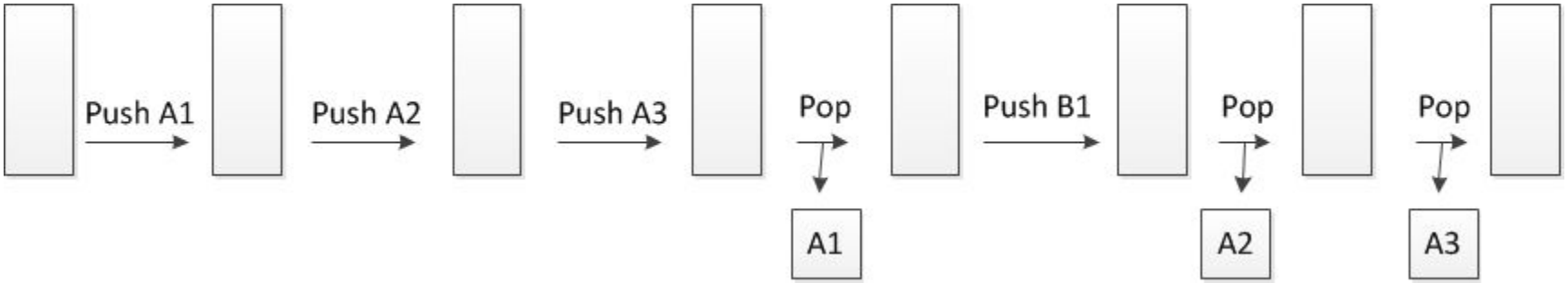
Очередь

- FIFO (First In – First Out)

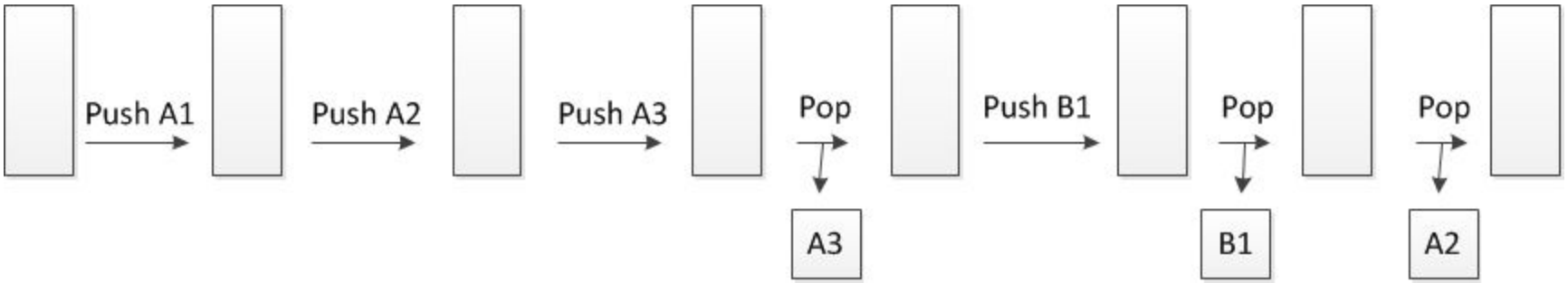
Стек

- LIFO (Last In – First Out)

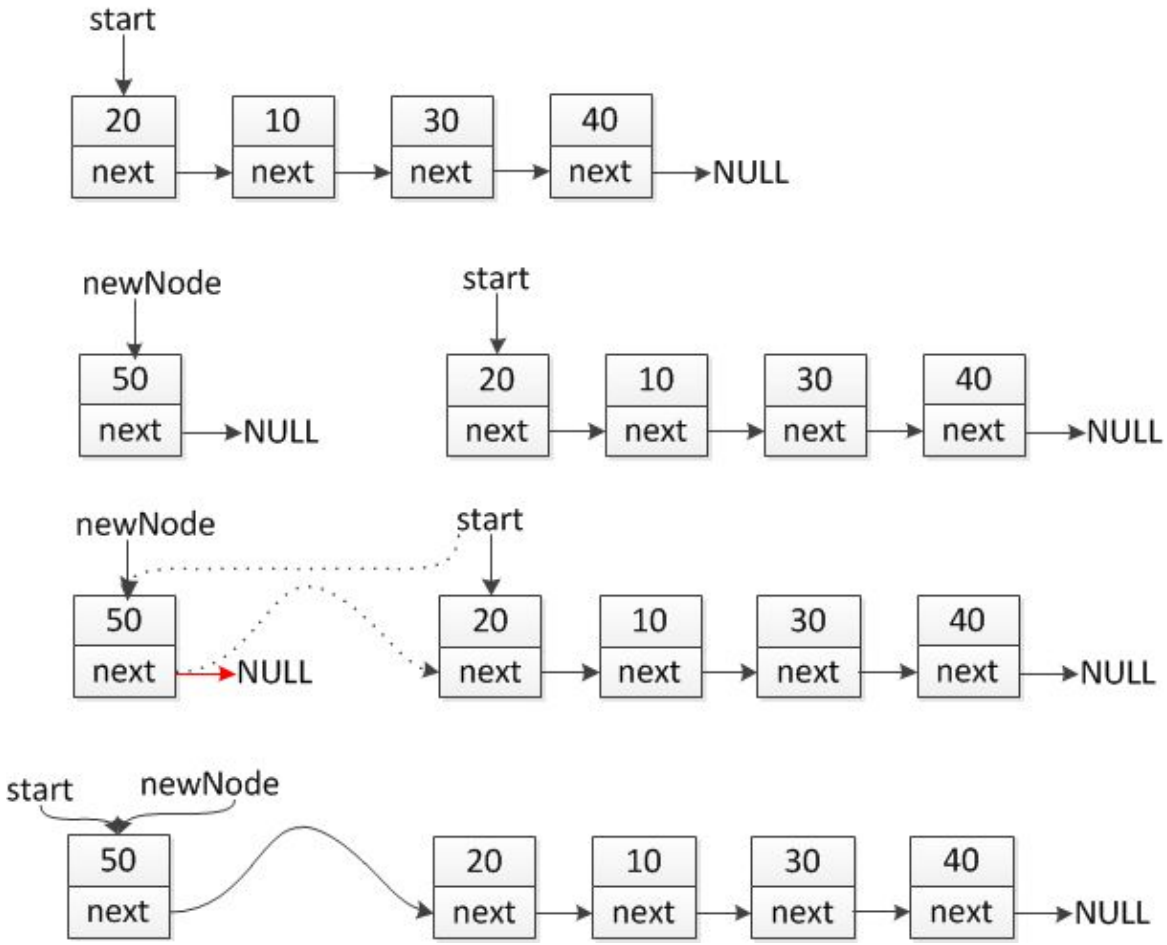
Очередь как черный ящик



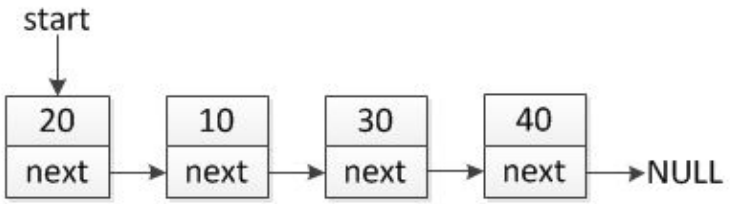
Стек как черный ящик

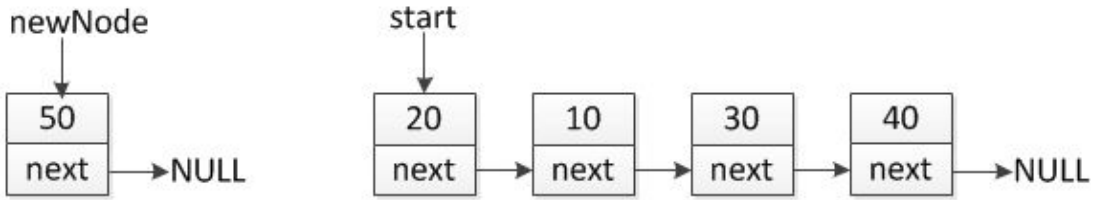


Дополнение стека



Дополнение списка

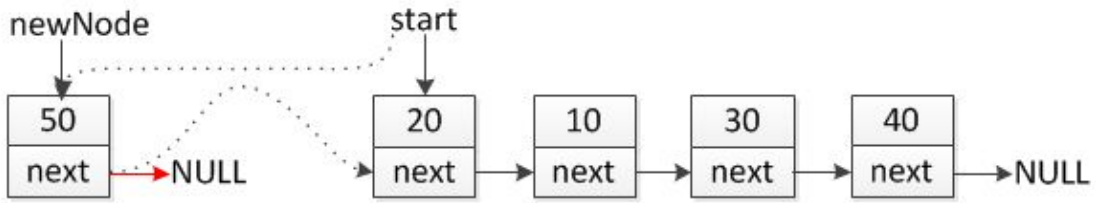




newNode = new node;

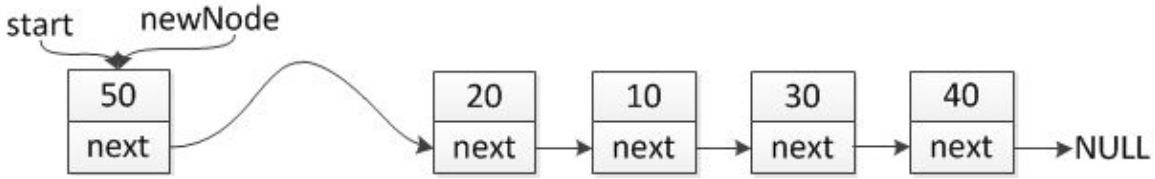
newNode->info = 50;

newNode->next = NULL;



newNode->next = start;

start =newNode;



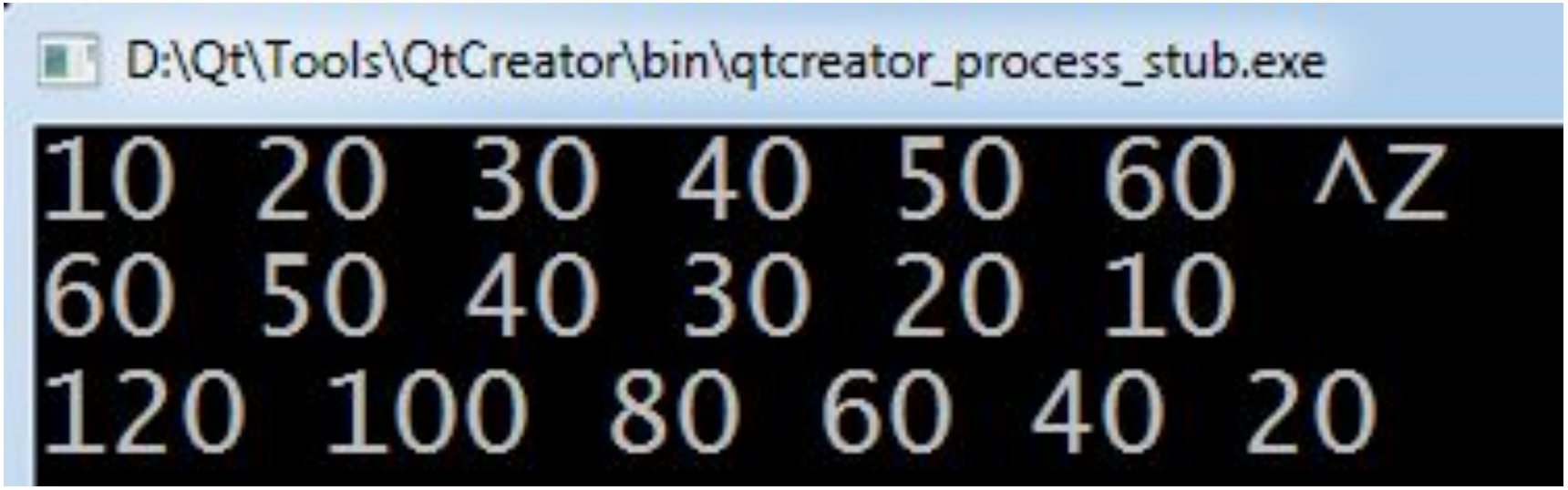
Ввод-обработка-вывод

```
node *head=NULL, *newNode; int x;
while(cin>>x){
	newNode = new node;
	newNode->info = x;
	newNode->next = NULL;
	if(head==NULL){
		head = newNode;
	} else {
		newNode->next = head;
		head = newNode;
	}
}
```

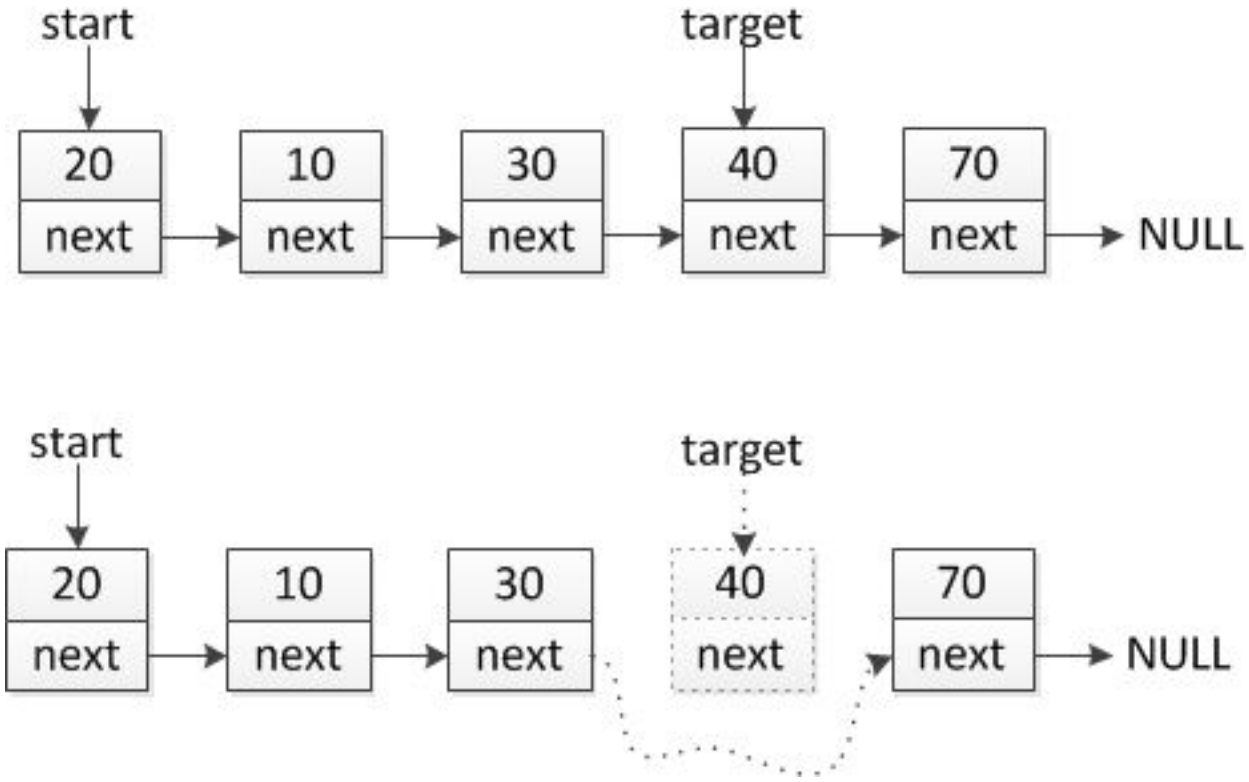
Ввод-обработка-вывод

```
for(node* cur = head; cur!=NULL; cur=cur->next){
	cout << cur->info << " ";
}
cout << endl;

for(node* cur = head; cur!=NULL; cur=cur->next){
	cur->info *= 2;
}

for(node* cur = head; cur!=NULL; cur=cur->next){
	cout << cur->info << " ";
}
cout << endl;
```

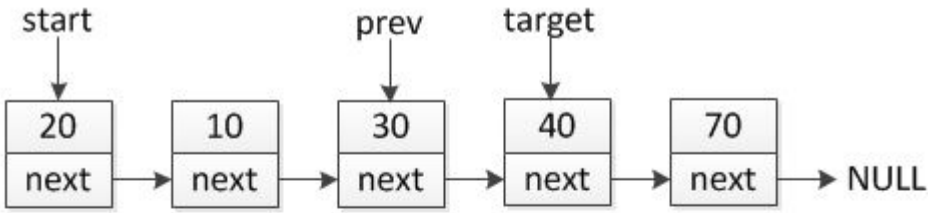
Результат работы

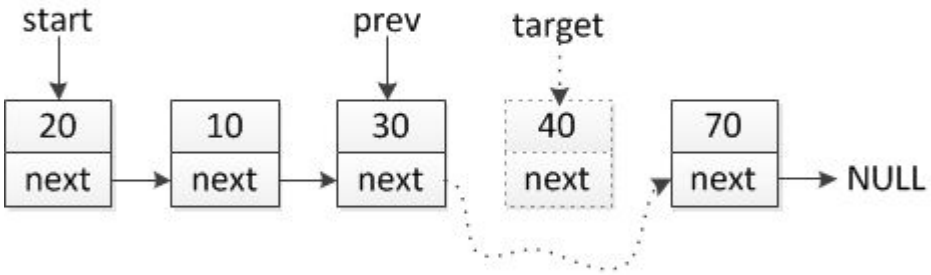


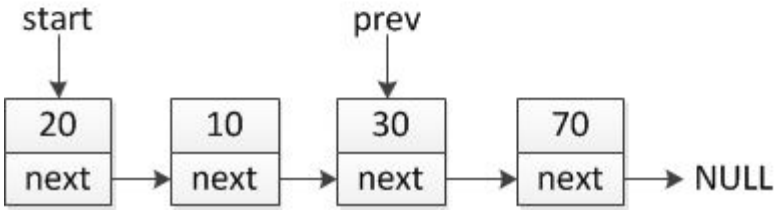
Удаление


start
target
20
next
10
next
30
next
40
next
70
next
NULL
start
target
20
next
10
next
30
next
40
next
70
next
NULL

Удаление

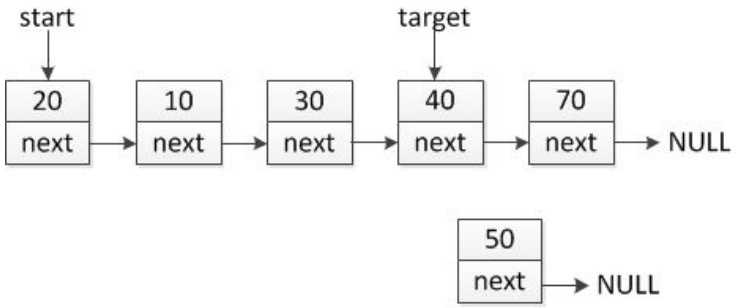


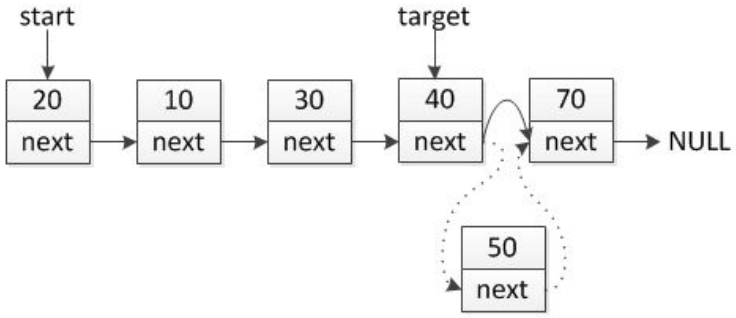


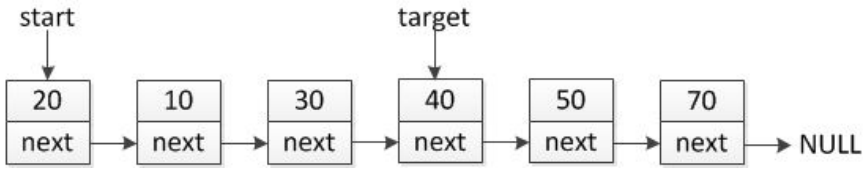



```
prev->next = target->next;
delete target;
```

Вставка в середину

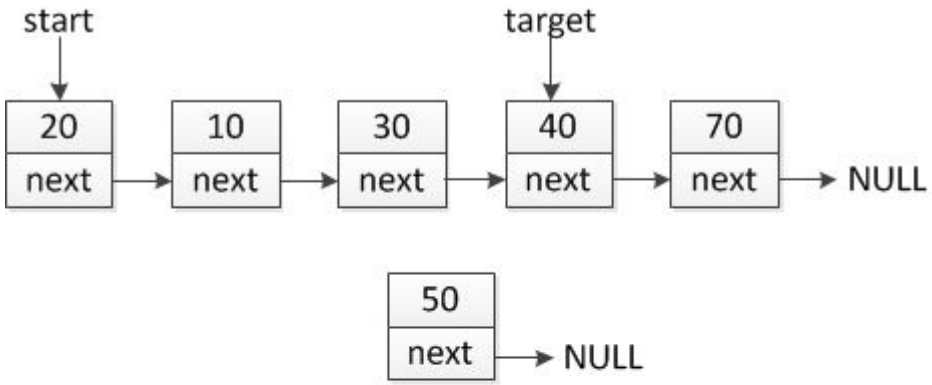


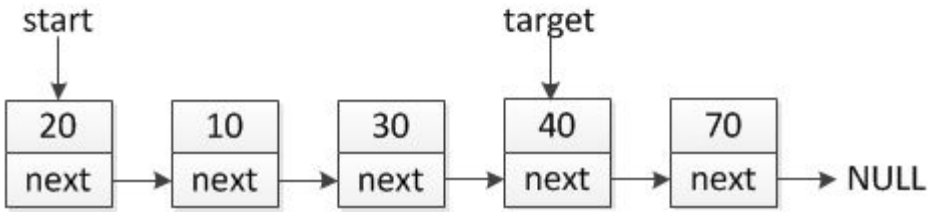




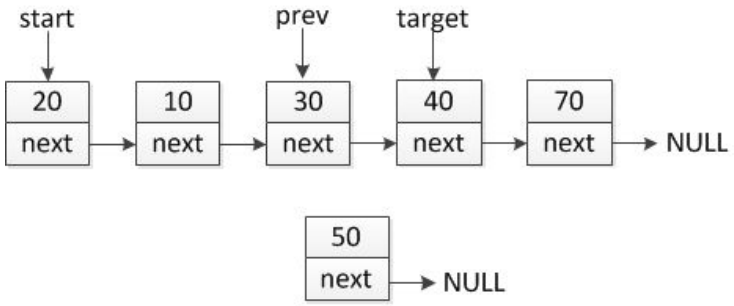

```
node newNode = new node;
node->info = 50;
node->next = NULL;

newNode->next = target->next;
target->next = newNode;
```

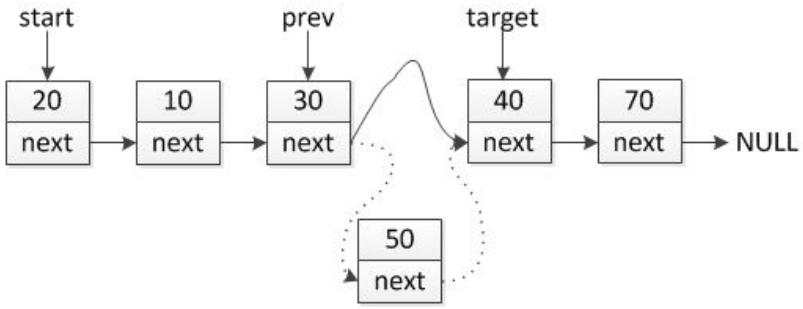
Вставка в середину

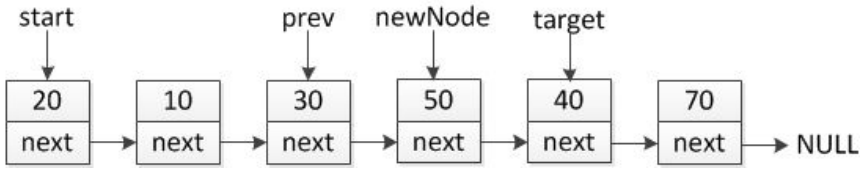




Вставка в середину








```
node newNode = new node;
node->info = 50;
node->next = NULL;

prev->next = newNode
newNode->next = target;
```