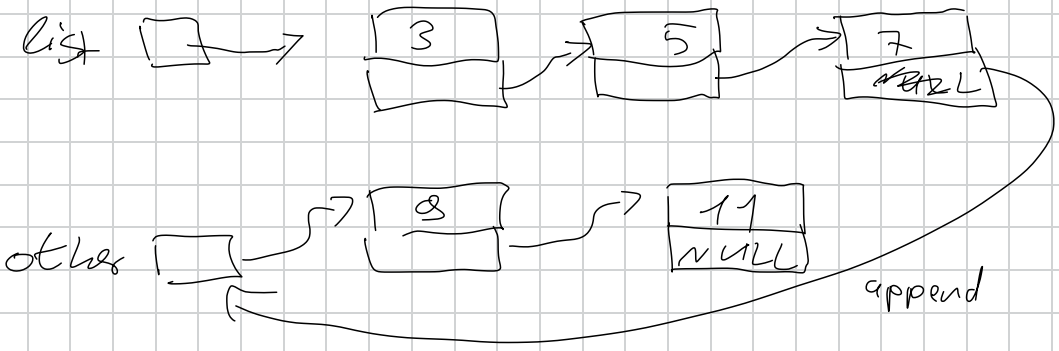
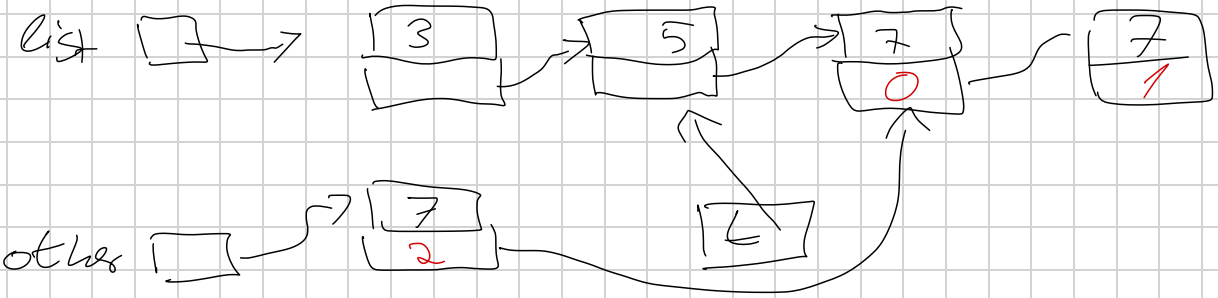
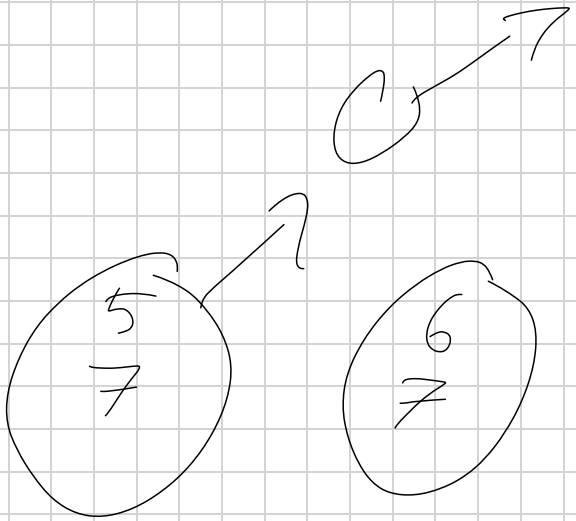
```
struct node {
    int num;
    struct node *next;
};
struct node *list;
```

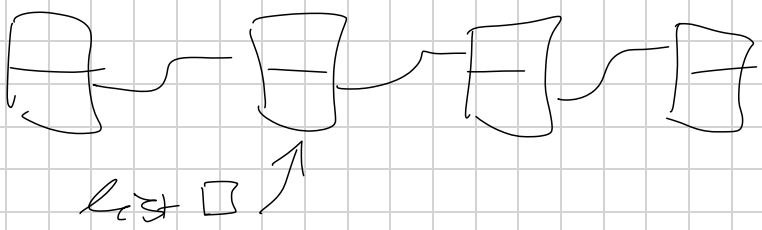



```
list = appn(list, other);

struct node *appnd(struct node *list,
                   struck node *rest){
struct node *e;
if (!list){
   list = rest;
} else {
   for(e = list; e->next; e=e->next)
          /*whee!*/;
   e->next = rest;
}
return list;
}
```



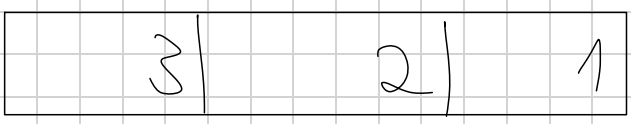

```
struct node *insert(struct node *list,
                    struct node *rest){
  struct node *t;

  if(!list || new->num <= list->num){
      new->next = list;
      list = new;
  } else {
      for(t = list; t->next && new->num > t->next->num; t = t->next);
          /* where */;
      new->next = t->next
      t->next = new;
  }
  return list;
```





```
struct node *append(struct node *list,
                    struct node *rest){

struct node head, *t;
head.next = list;
for(t=&head; t->next && t->next->num > new->next; t=t->next)
        /*nothing*/;
    new->next = t->next
    t->next = new;
    return list;
}
```

```
int A[] = {3,2,1};
```



```
void qsort(void *base, size_t nmemb, size_t size,
           int (*compar)(const void *ap, const void *bp));
```

```
qsort(A, sizeof(A)/sizeof(int), sizeof(int), bigger);
```

```
int bigger(const void *ap, const void *bp){
    int a = *(int *)ap;
    int b = *(int *)bp;
    return a - b;
}
```