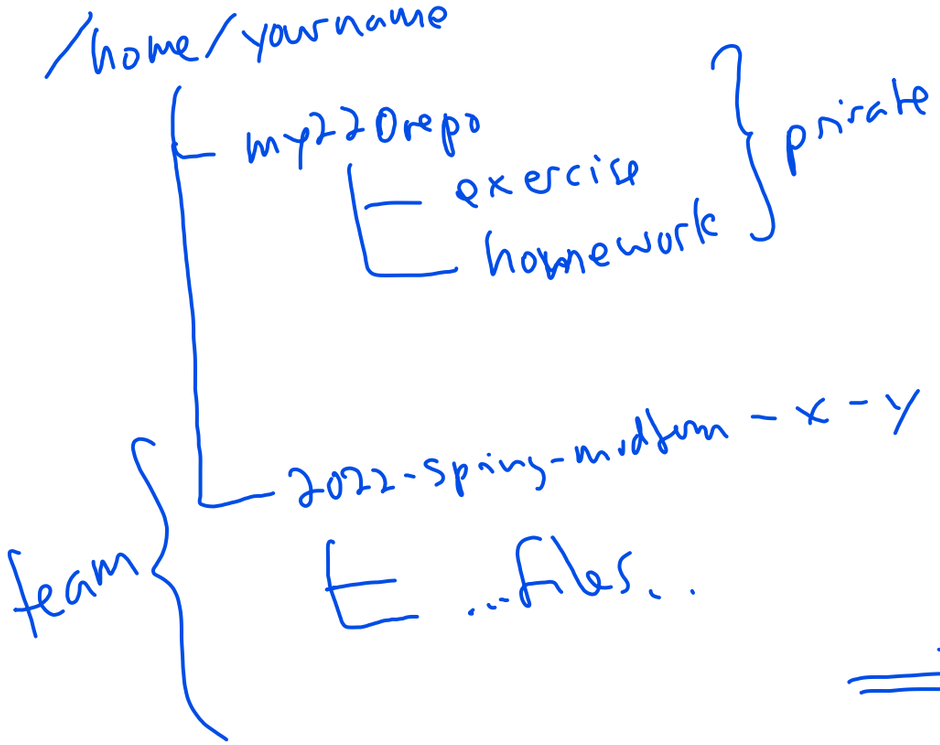
Day 19 (Mon 03/07)

- exercise 18 review
- work on midterm project / exercises



Announcements/reminders

- Last chance to register your midterm project team: by 11:59 pm this evening
  - If you are not registered as being on a team, we will assign you to a team

- Midterm exam: in class Friday 03/11
  - computer based
    - we recommend using lab PCs
    - if you choose to use your own laptop, you accept the risk of hardware issues, network issues, etc.

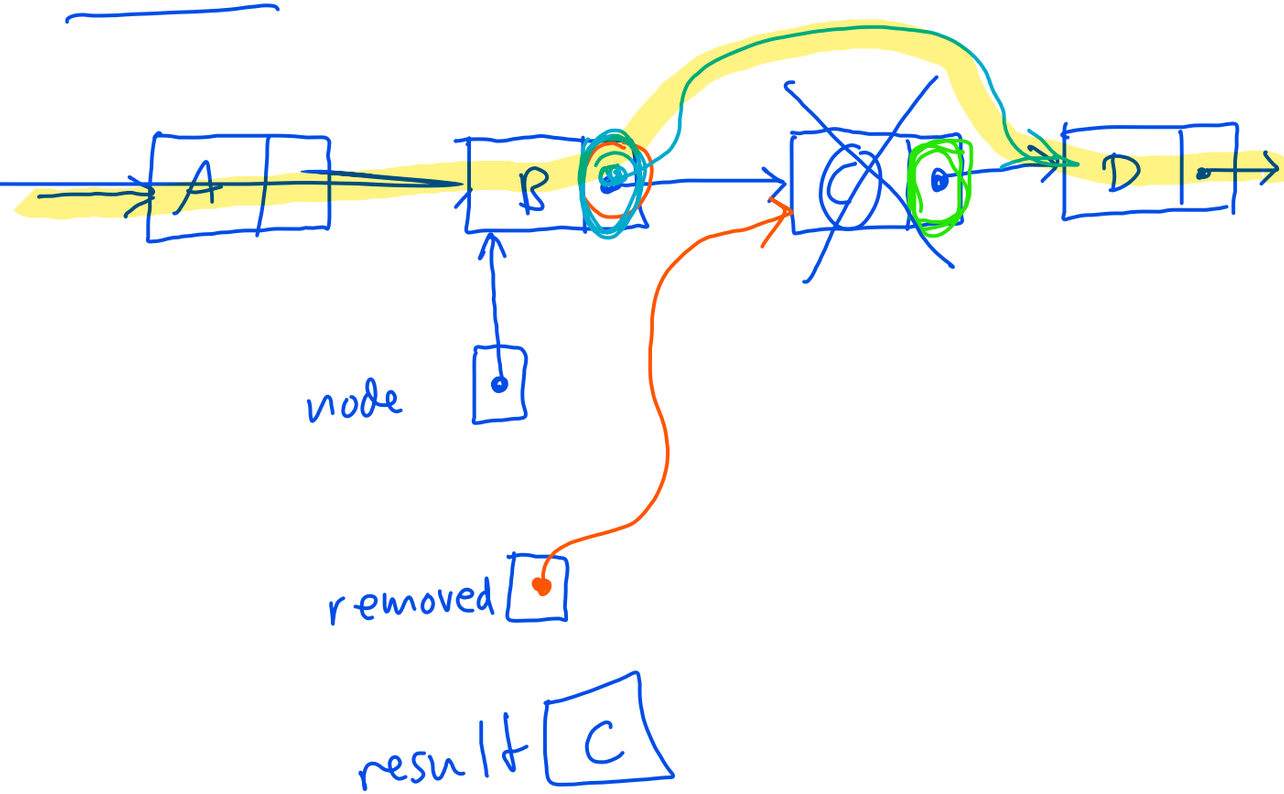
- Midterm project: due Friday 03/18 by 11pm
  - Late submissions are not accepted

Exercise 18 review

```
char remove_after(Node * node) {
  Node *removed = node->next;
  if (removed == NULL) {
    return '?';
  }

  node->next = removed->next;
  char result = removed->data;
  free(removed);
  return result;
}
```

Trace:



Exercise 18 review

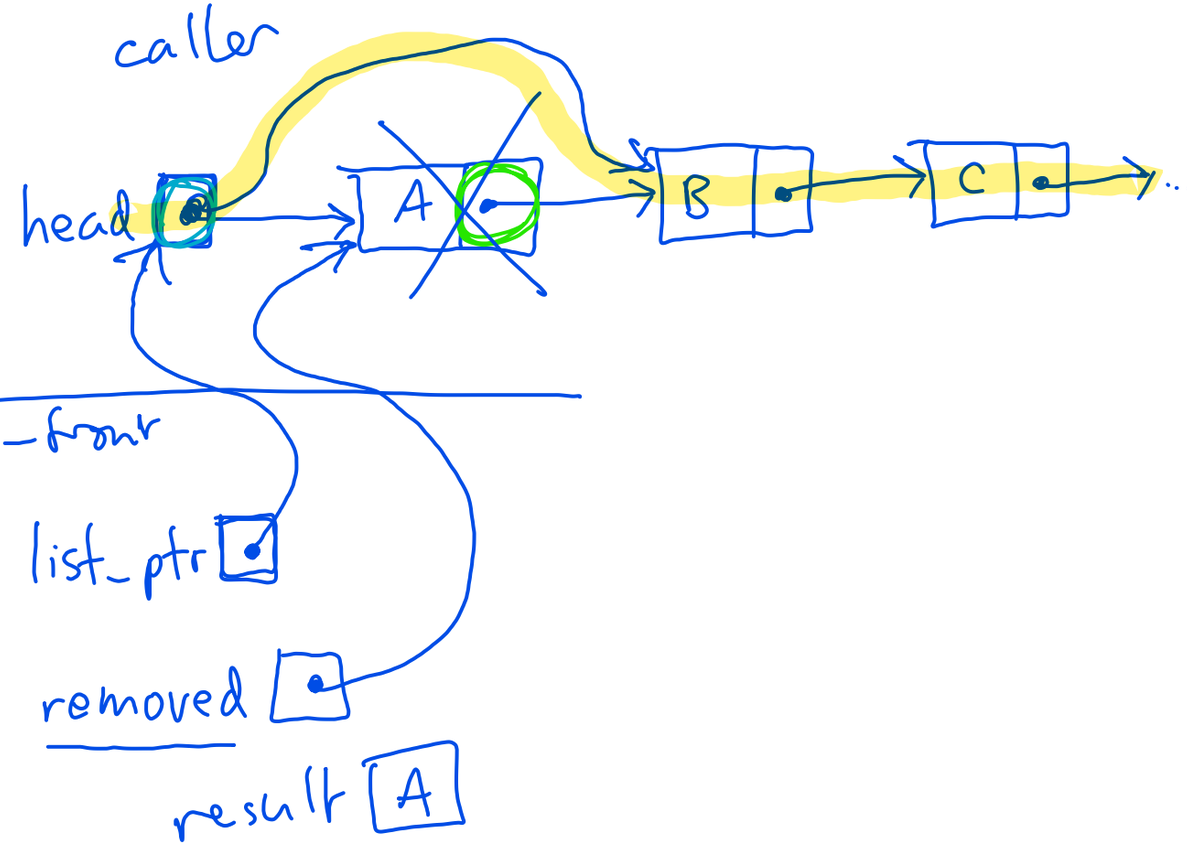
```
char remove_front(Node ** list_ptr) {
  if (*list_ptr == NULL) {
    return '?';
  }

  Node *removed = *list_ptr;
  *list_ptr = removed->next;
  char result = removed->data;
  free(removed);
  return result;
}
```

head

head

Trace:



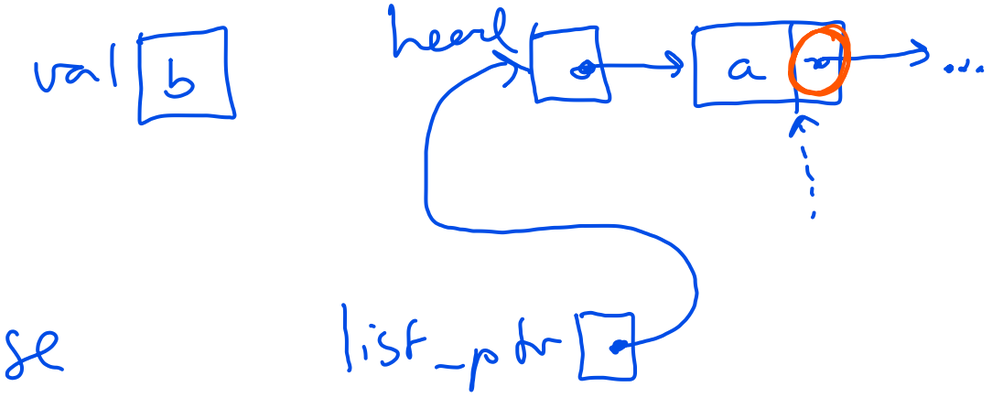
equivalence

*list_ptr

Same as

head

Exercise 18 review




```
void remove_all(Node ** list_ptr, char val) {
  if (*list_ptr == NULL) {
    return; // reached end of list
  }
  if ((*list_ptr)->data == val) {
    // remove first element
    remove_front( list_ptr );

  } else {
    // continue on rest of list
    list_ptr = & (*list_ptr)->next;
  }
  remove_all(list_ptr, val);
}
```

base case

remove lst elt

or

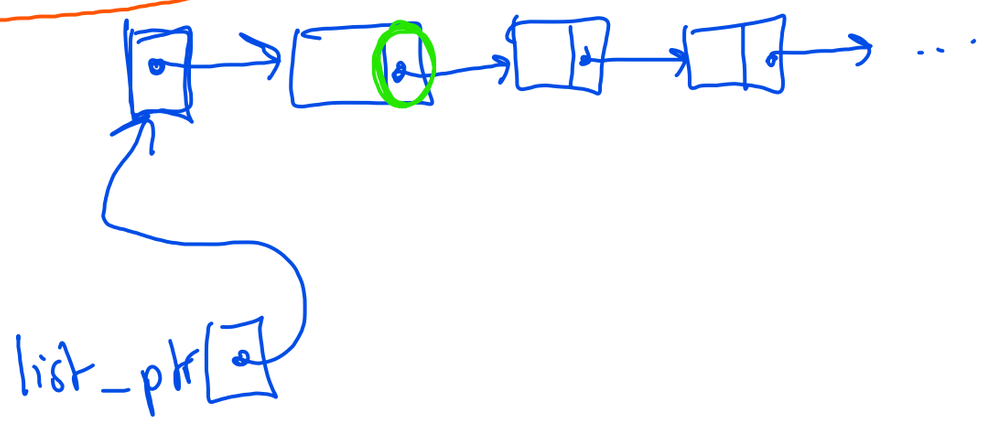
skip lst elt

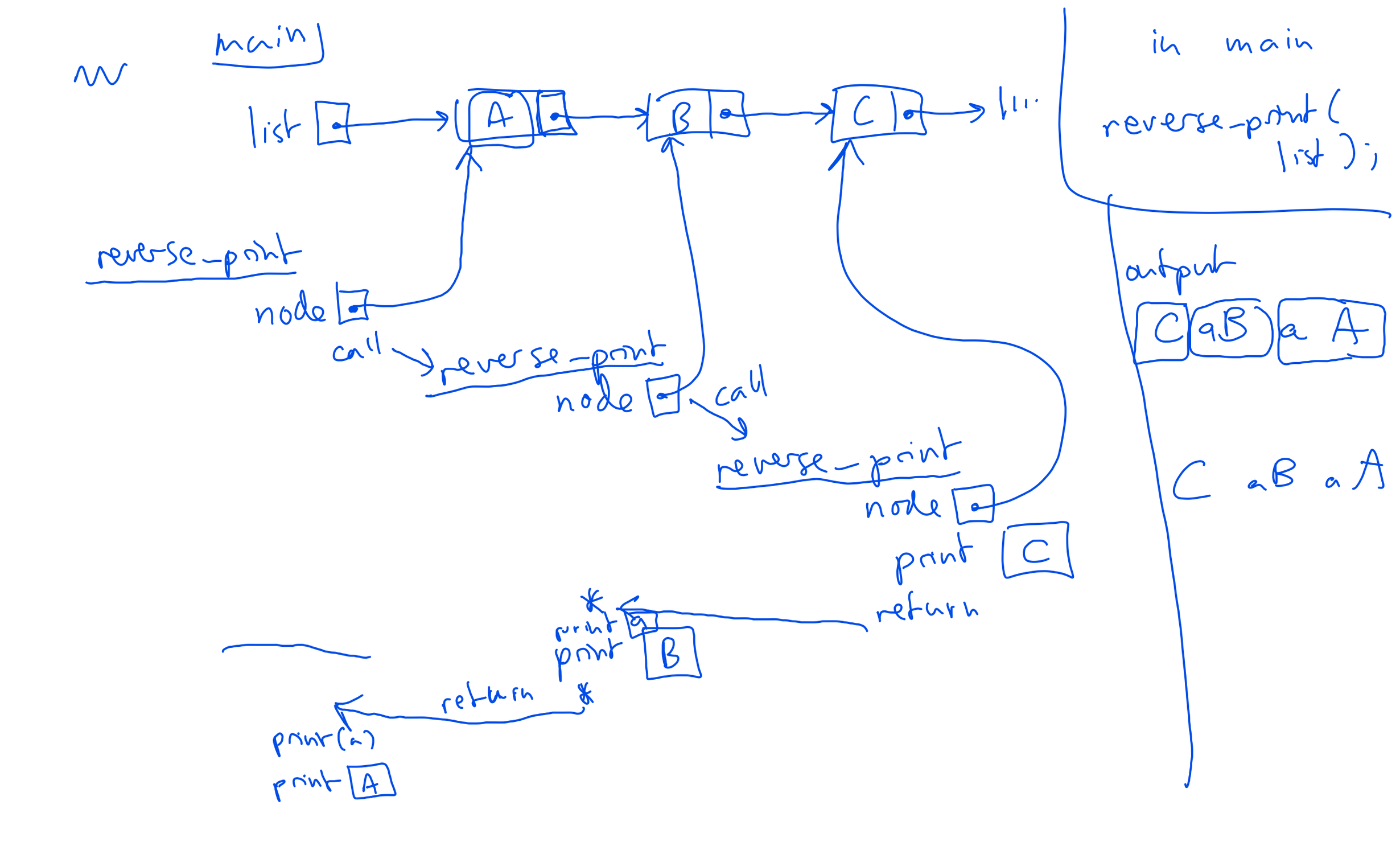
pointer to pointer var pointing to rest of list

Exercise 18 review

```
Node * insert(Node ** list_ptr, char val) {
  if (*list_ptr == NULL || val < (*list_ptr)->data) {
    add_front(list_ptr, val);
    return *list_ptr;
  } else {
    // recursion
    return insert(&(*list_ptr)->next, val);
  }
}
```

base cases: empty list, or val is less than the value of the first node




main
list
A
B
C
...
reverse_print
node
call
reverse_print
node
call
reverse_print
node
print
C
return
print
B
return
print(a)
print
A
in main
reverse_print(
list);
output
C
aB
a A
C aB a A