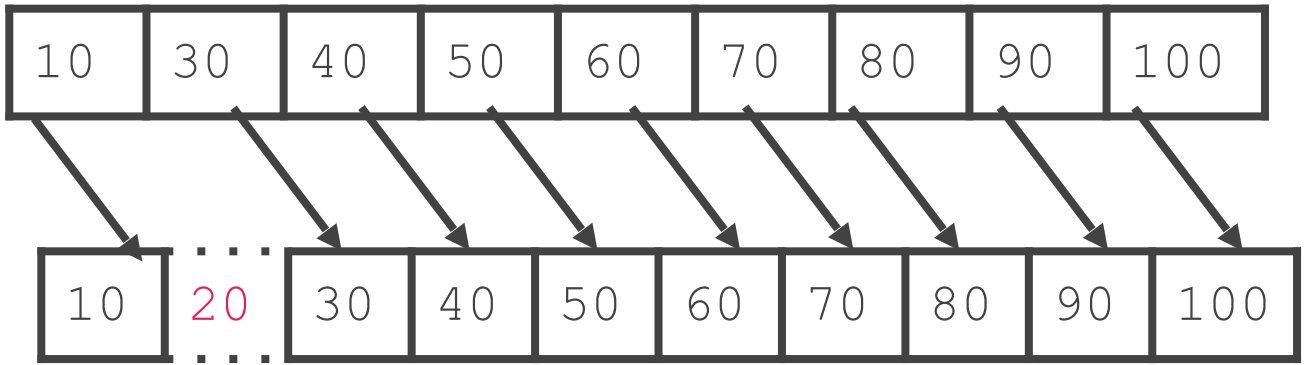
Lecture 5

Linked List

Inserting a number in an array

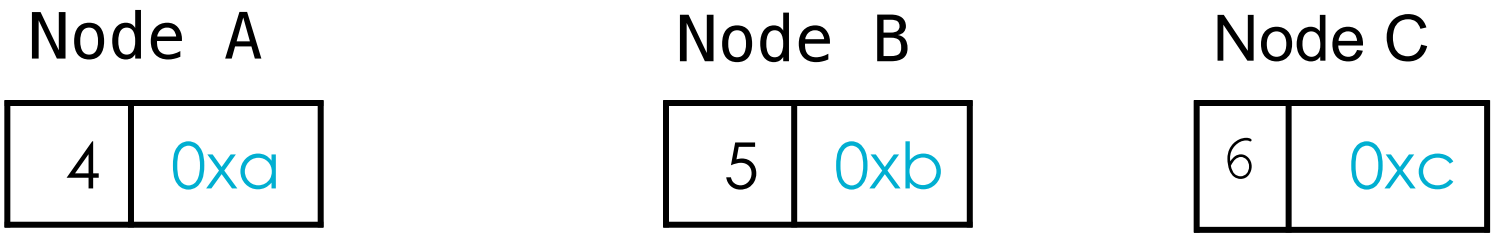
20

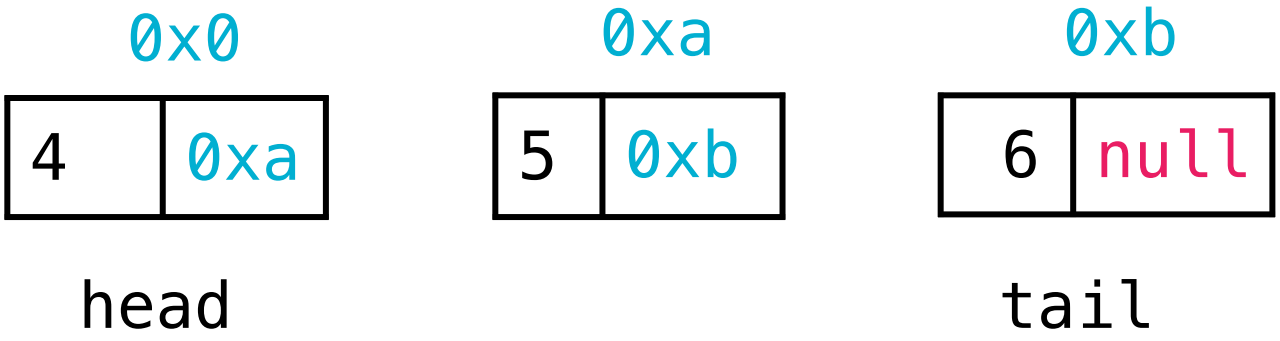
10 ^ 30 40 50 60 70 80 90 100

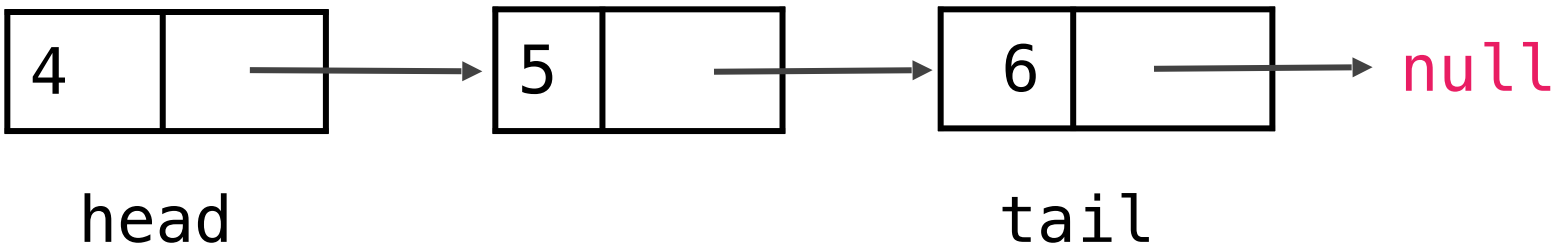


Singly linked list

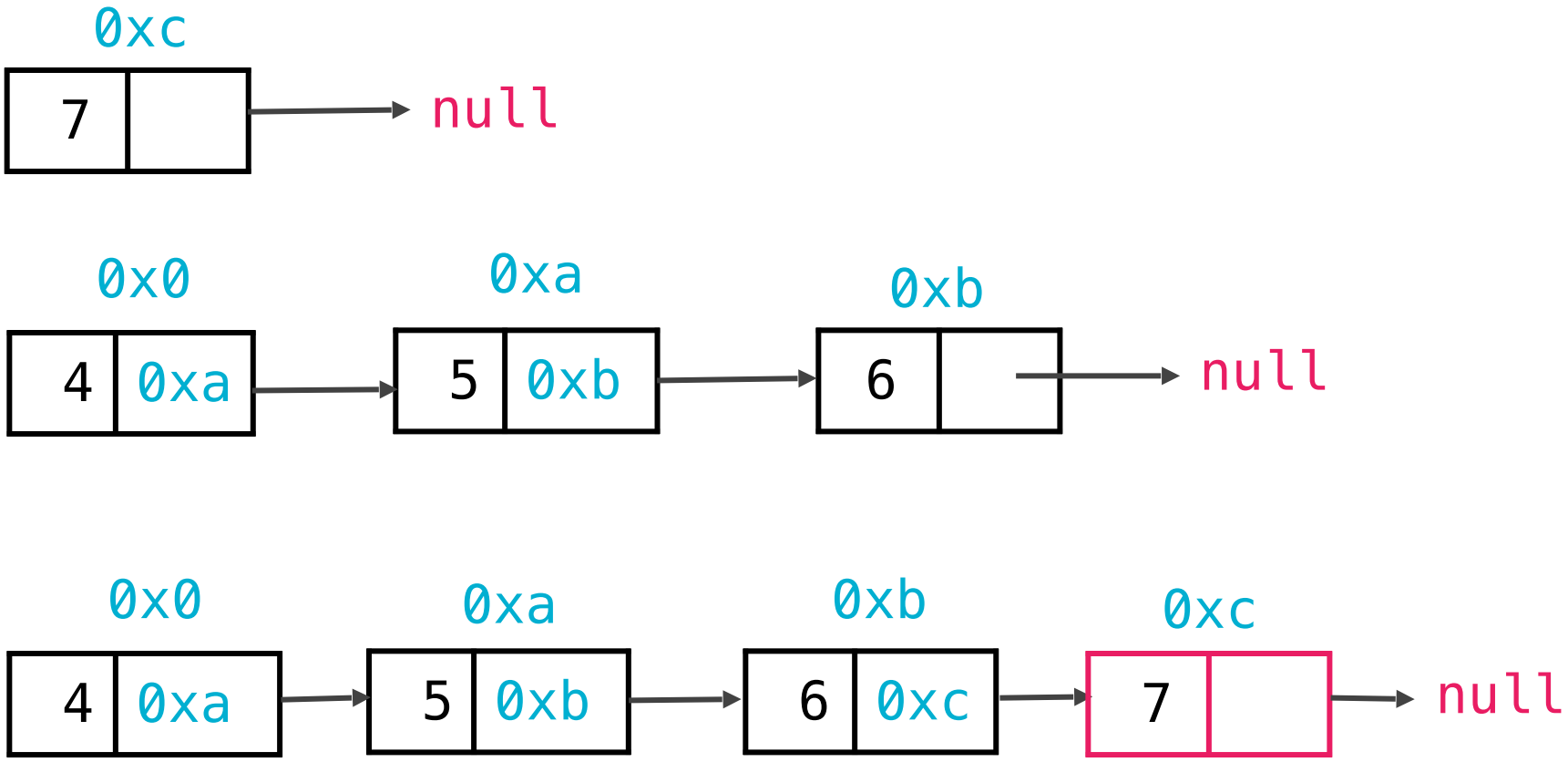
```
struct node{
  int data;
  node* next;
};
```




0x0
4
0xa
0xa
5
0xb
0xb
6
null
head
tail


4
5
6
null
head
tail

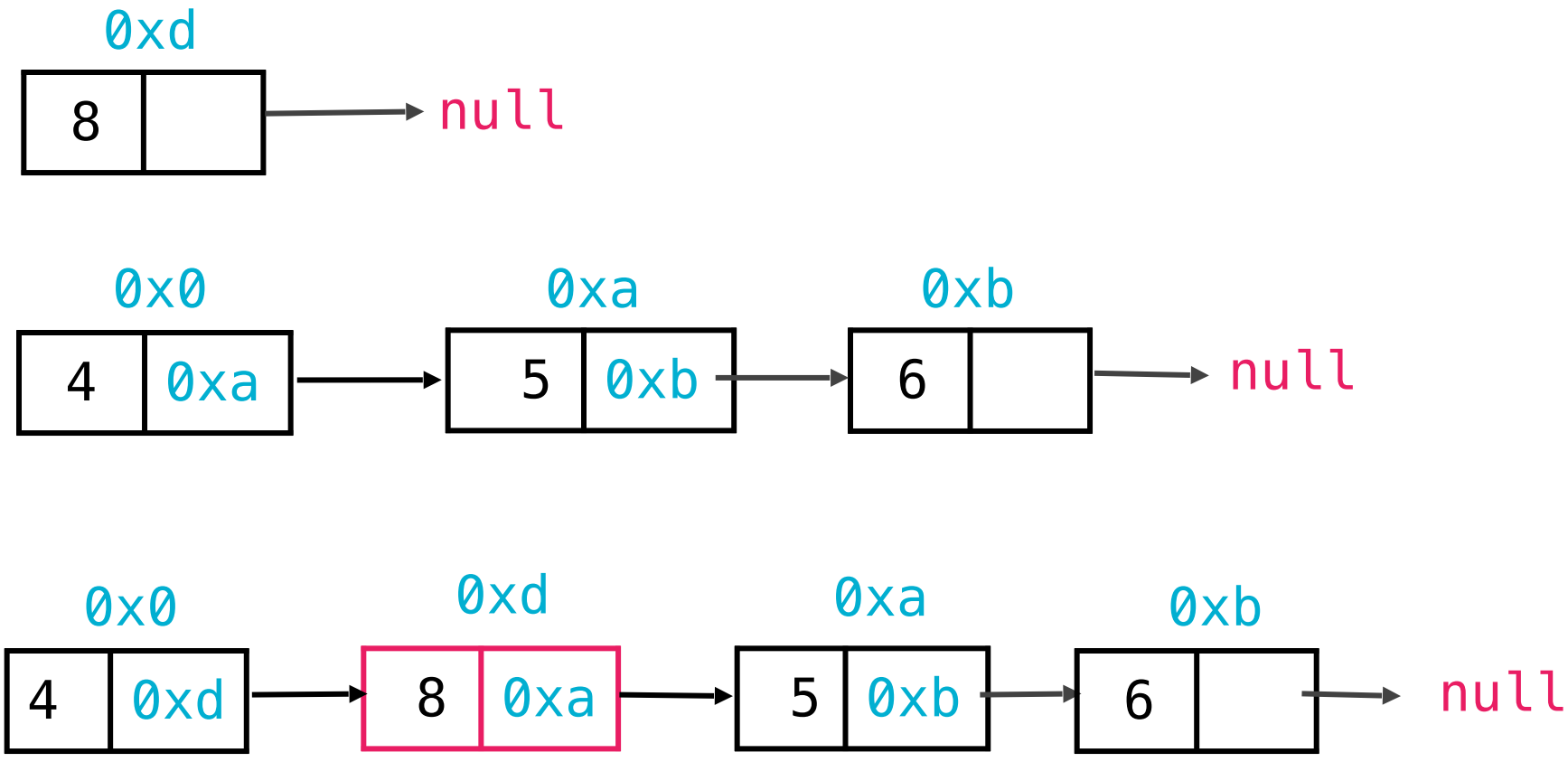
add an element at the end of the list


0xc
7
null
0x0
4
0xa
0xa
5
0xb
0xb
6
null
0x0
4
0xa
0xa
5
0xb
0xb
6
0xc
0xc
7
null

```
void add(node* &sll, int data){
        if(sll == NULL){
                node *newNode = new node;
                newNode->data = data;
                newNode->next = sll; //i.e equal to NULL
                (sll) = newNode;
        }
...
```

```
...
        else{
                node *current = sll;
                while(current->next!=NULL){
                        current = current->next;
                }
                node *newNode = new node;
                newNode->data = data;
                newNode->next = current->next;
                current->next = newNode;
        }
}
```

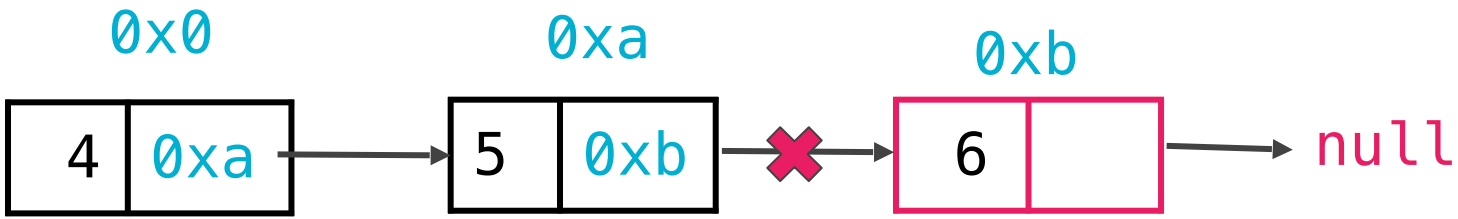
add an element anywhere in the list

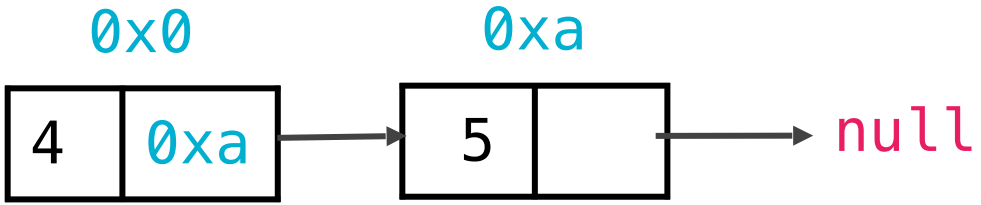

0xd
8
null
0x0
4
0xa
0xa
5
0xb
0xb
6
null
0x0
4
0xd
0xd
8
0xa
0xa
5
0xb
0xb
6
null

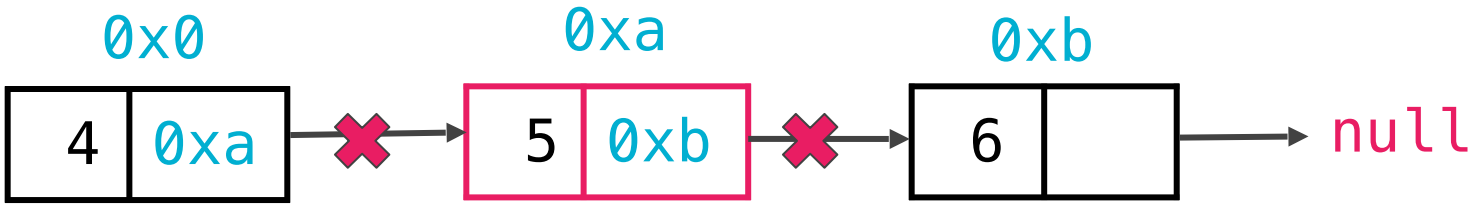
```
void insert(node* &sll, int index, int data){
    if(index == 0){
        node* newNode = new node;
        newNode->data = data;
        newNode->next = sll;
        sll = newNode;
    }
...
```

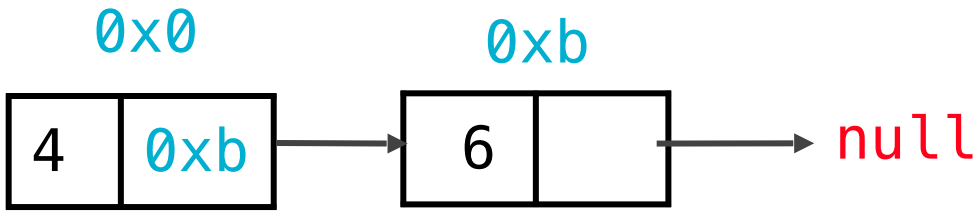
```
...
else{
        node* current = sll;
        for(int i =0; i<index-1; ++i){
                current = current->next;
        }

        node* newNode = new node;
        newNode->data = data;
                newNode->next = current->next;
                current->next = newNode;
        }
}
```

remove an element from anywhere in the list


0x0
0xa
0xb
4
0xa
5
0xb
6
null


0x0
0xa
4
0xa
5
null


0x0
0xa
0xb
4
0xa
5
0xb
6
null


0x0
0xb
4
0xb
6
null

```
void removeElement(node* &sll, int index){
      if(index ==0){
            node* junk = sll;
            sll = sll->next;
            delete junk;
      }
...
```

```
...
        else{
                node* current = sll;
                for(int i =0; i<index-1; ++i){
                        current = current->next;
                }
                node *junk = current->next;
                current->next = current->next->next;
                delete junk;
        }
}
```

print the list

```
void display(node *&list) {
  node* current = list;
  while (current!= NULL){
     cout<<" "<< current->data<<" ";
     current = current->next;
  }
}
```