CSC 215

Math and Computer Science

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

View as Separate Items (node)

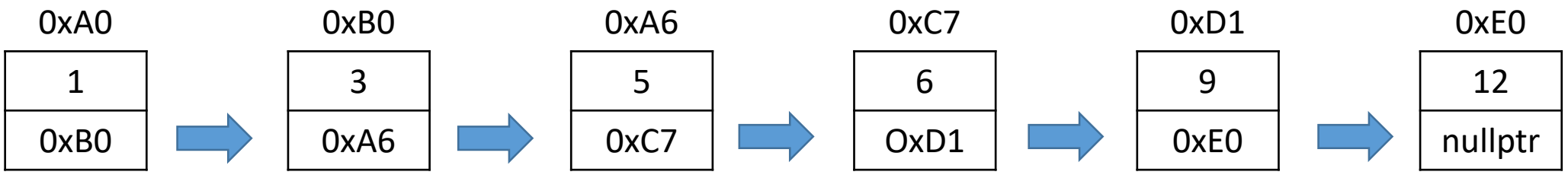
1	3	5	6	9	12

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Store address of next node

0xA0	0xB0	0xA6	0xC7	0xD1	0xE0
1	3	5	6	9	12
0xB0	0xA6	0xC7	OxD1	0xE0	

A Linked List



Last node will contain the value nullptr to indicate the end of the list.

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Node Structure

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

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

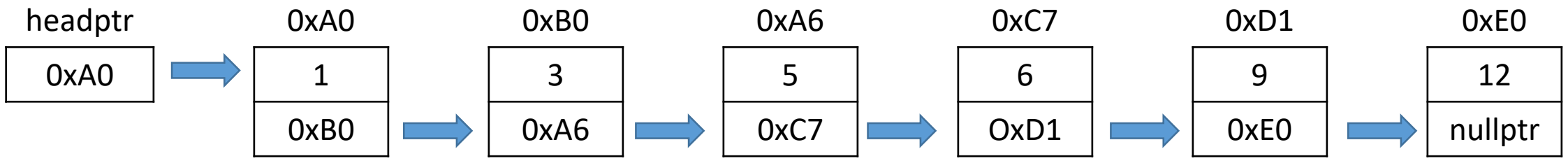
Where is the first node

- Must have a way to access the first element
- This is usually call headptr in textbooks
- It is just a pointer to the first node in the list
- Set to nullptr to indicate an empty list

```
node *headptr = nullptr;
```

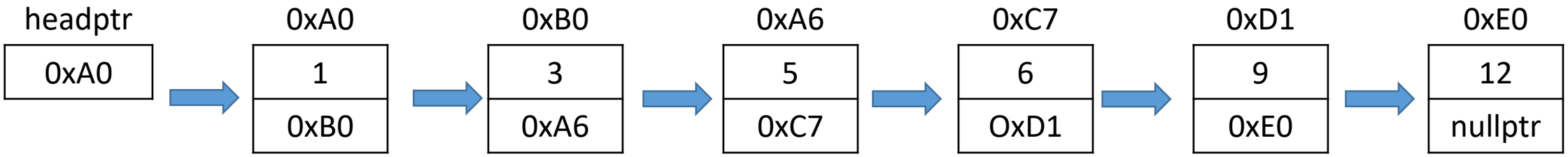
SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

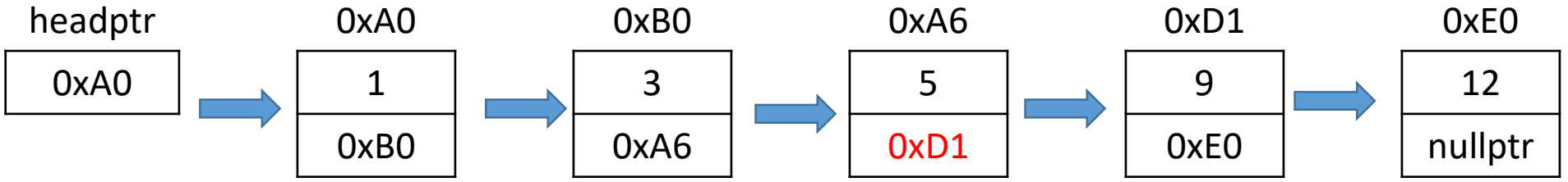
A Linked List



SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

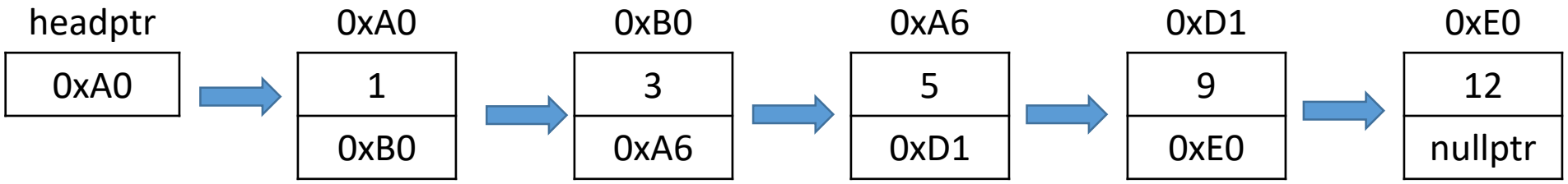
The list Shrinks – remove 6

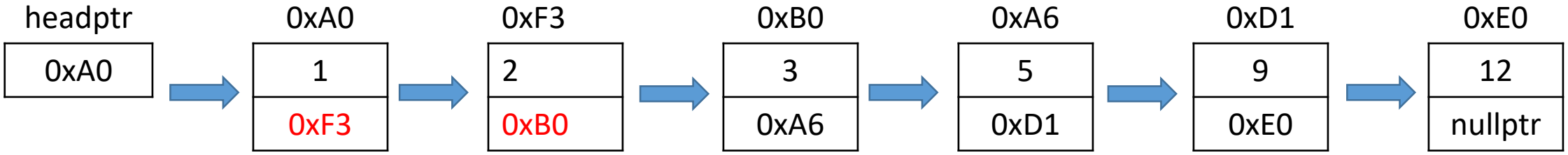




SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

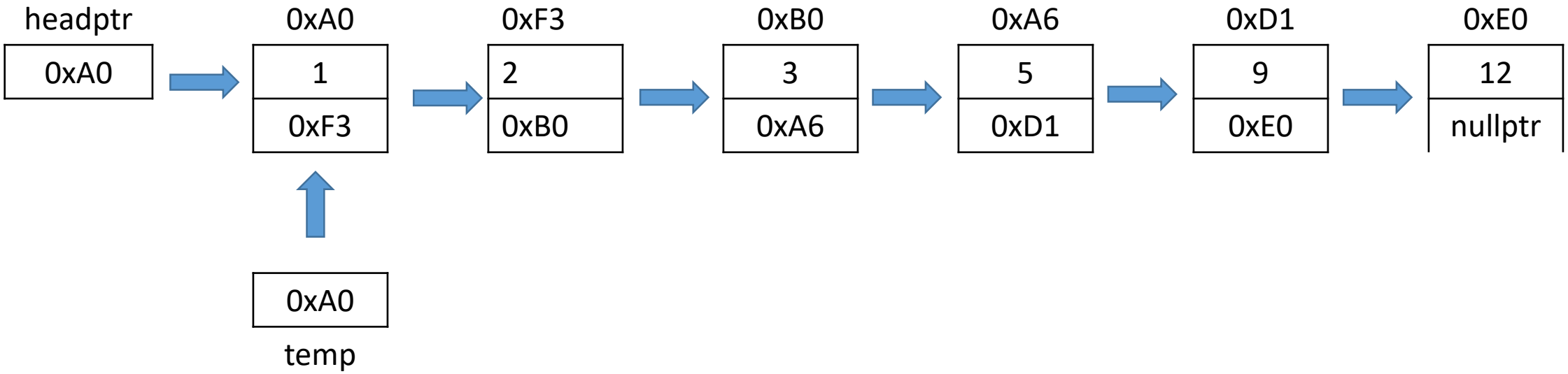
The list Grows – Insert 2





SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Walking the list

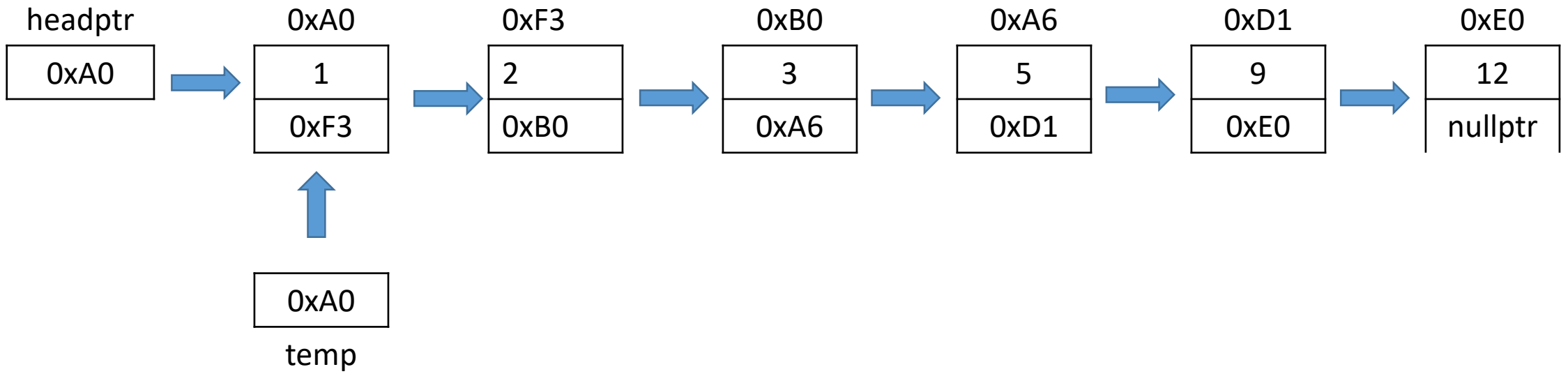



```
node *temp = nullptr;
//copy headptr into temp;
temp = headptr;
```

Use a temporary pointer (temp). Never change headptr unless absolutely necessary

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

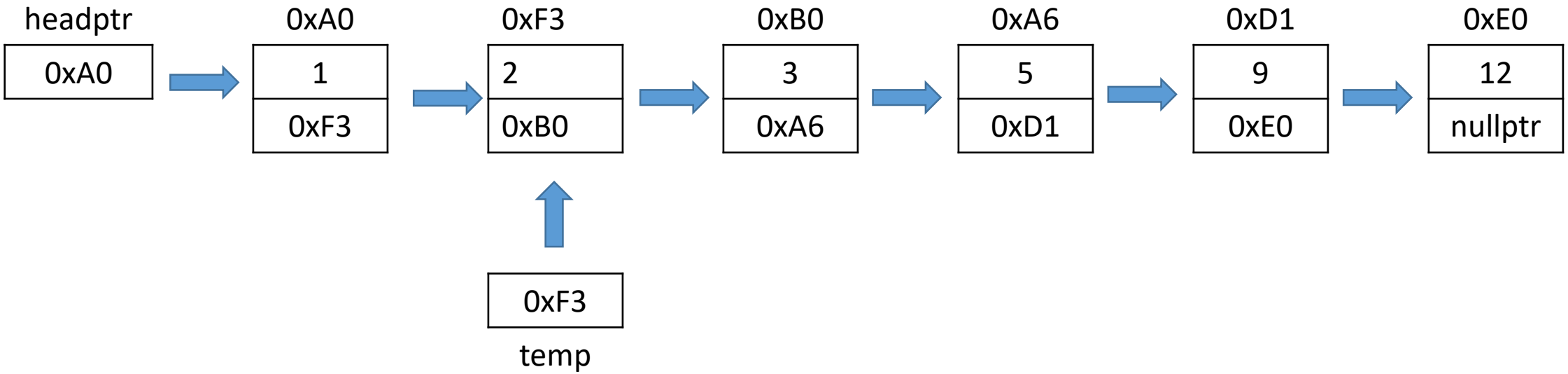
Walking Through the List



Output:
1

Out item and move temp down

Walking the list

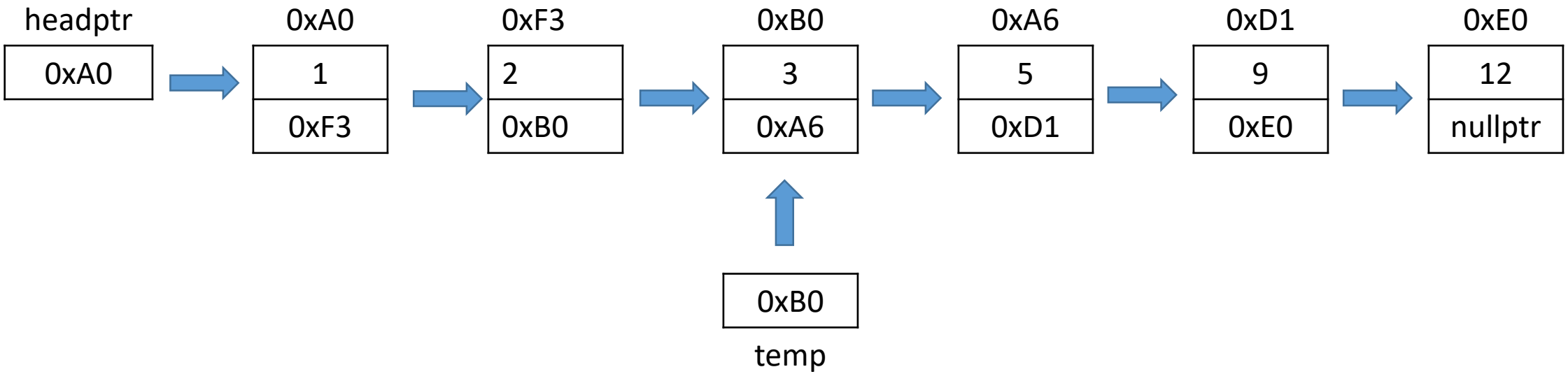


Output:
1 2

Output item and move temp down

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Walking the list

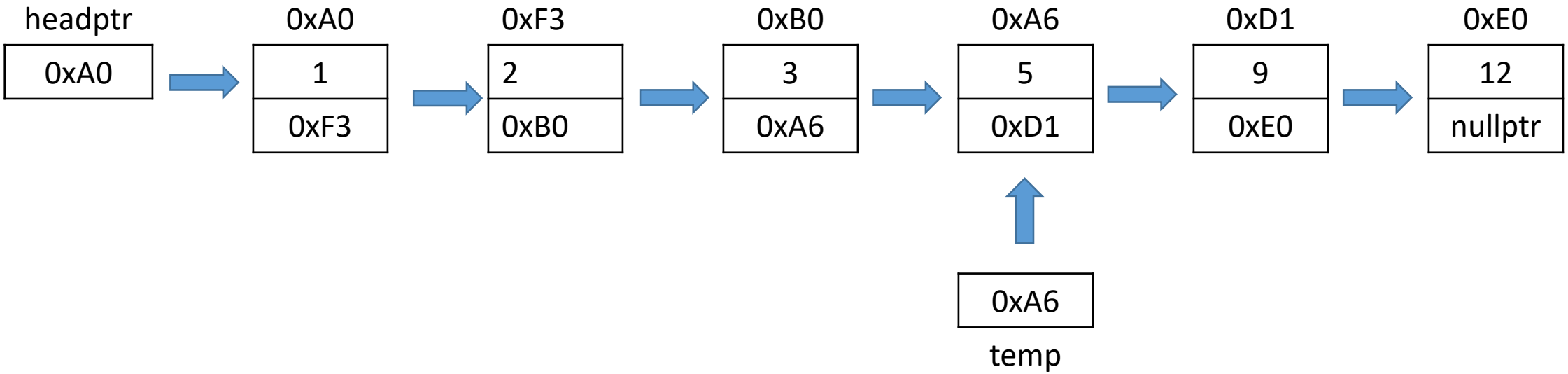


Output:
1 2 3

Output item and move temp down

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Walking the list

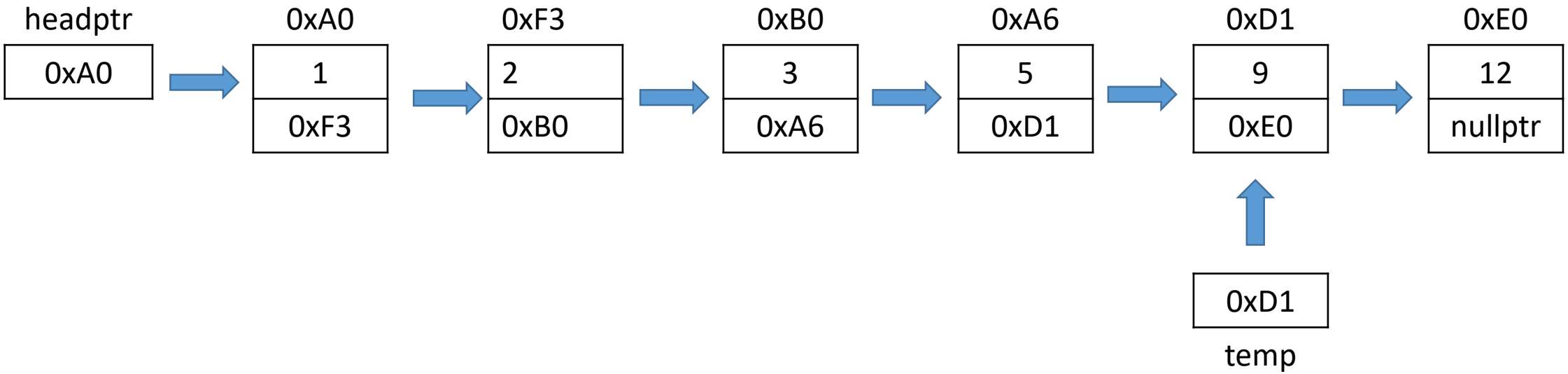


Output:
1 2 3 5

Output item and move temp down

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Walking the list

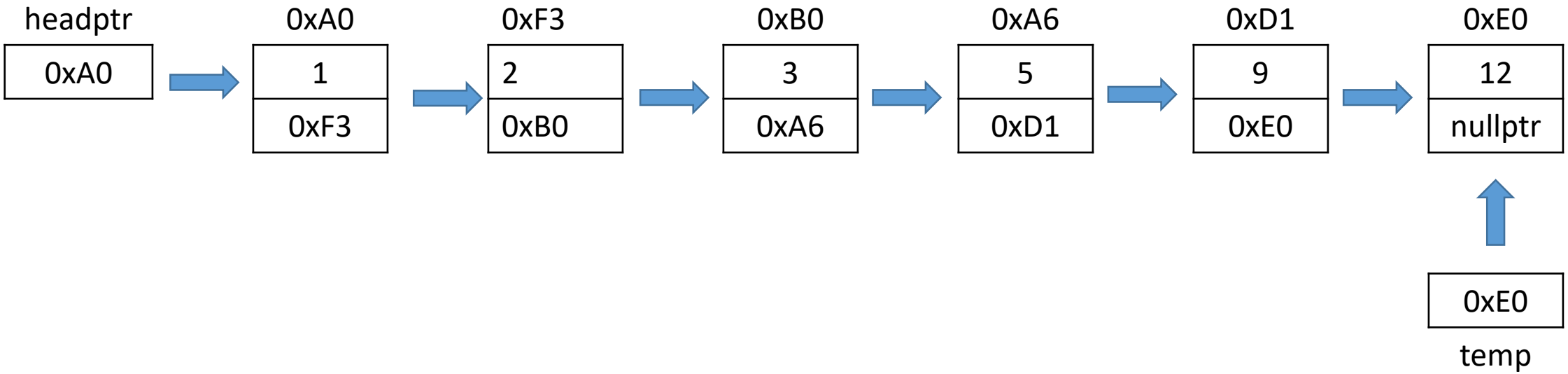


Output:
1 2 3 5 9

Output item and move temp down

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Walking the list

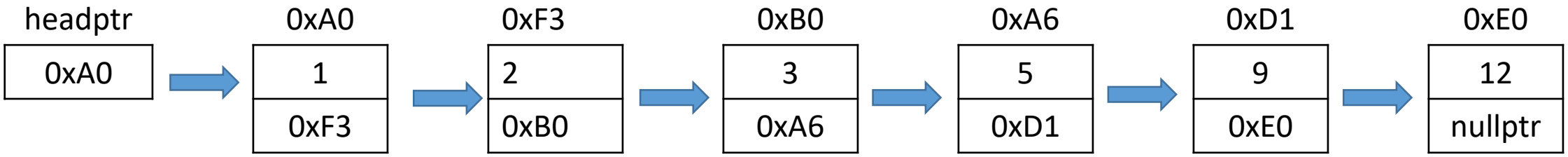


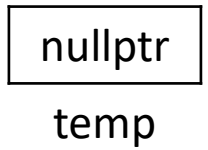
Output:
1 2 3 5 9 12

Output item and move temp down

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Walking the list





Output:
1 2 3 5 9 12

Temp has the value nullptr quit.

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Walking the Linked List

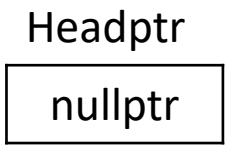
- This concept is used for many functions
  - Find
  - Count items
  - Print

Building the List

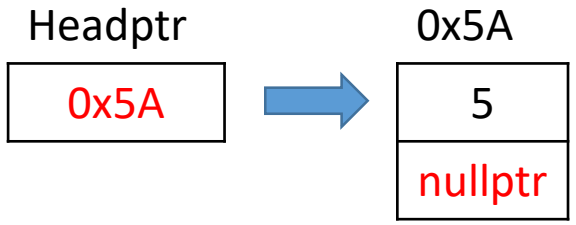
- Must handle insertions into
  1. Empty List
  2. Beginning of the List
  3. Middle of the List
  4. End of the list

Empty List Insertion

- Add something into the list to make it valid



- After adding a node, it must be a valid list

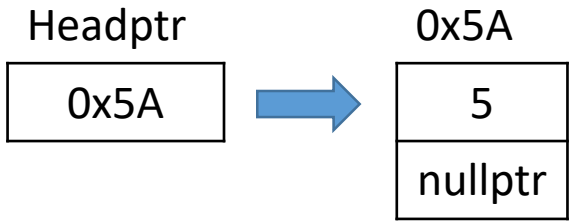


- Notice the last node has a nullptr and headptr now has address of the first node

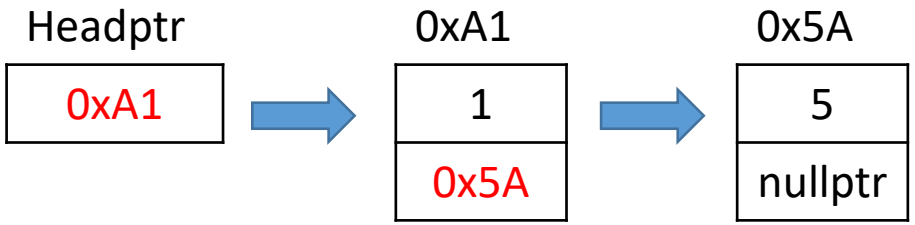
SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Beginning of the List Insertion

- Add something to the front of the list



- After adding a node, it must be a valid list

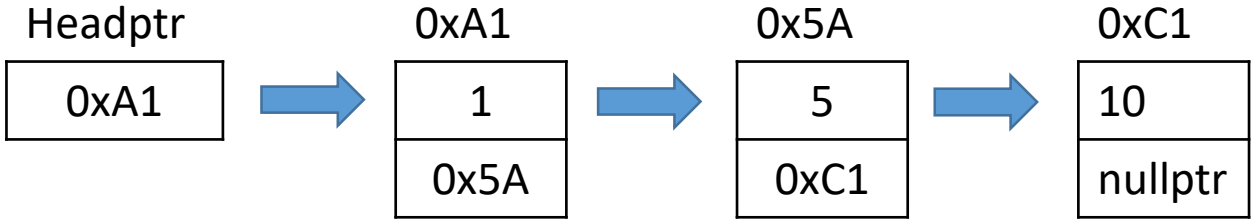


- First node point to the old of list
- Headptr has the address of the new node at the front of the list

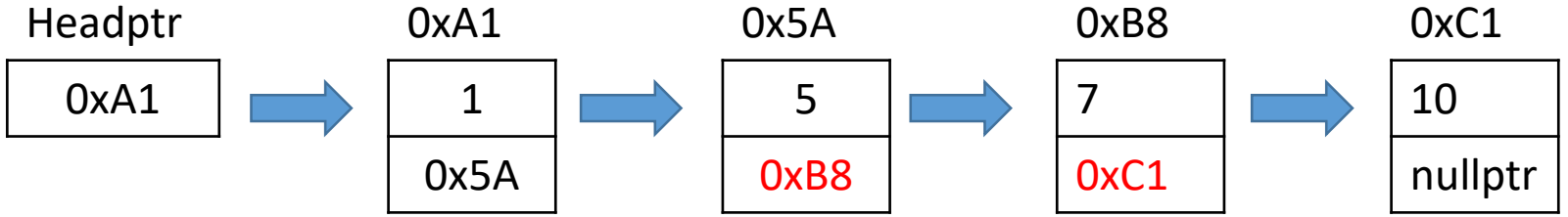
SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Middle of the List Insertion

- Add something to the middle of the list



- After adding a node, it must be a valid list



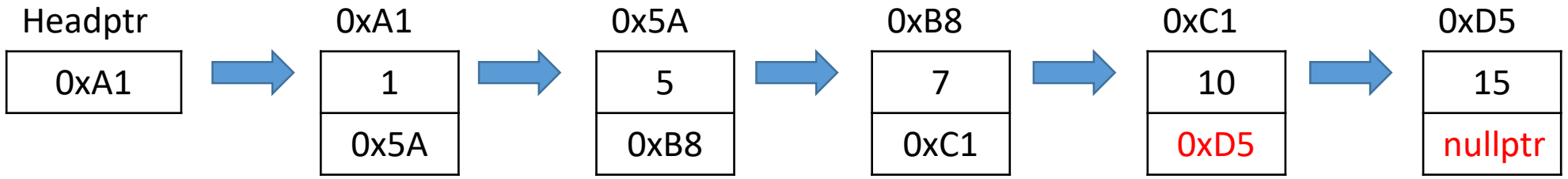
- New node contains the address of the remain list
- Node prior to the new node has address of the new node

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

End of the List Insertion

- Add something to the middle of the list

- After adding the node, it must be a valid list



- Old end of list points to new node
- New end of list has nullptr to show no more list

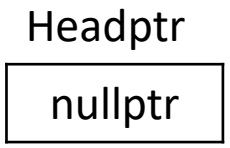
SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Removing from the List

- Must handle deletions from
  1. Empty list
  2. Beginning of List
  3. Middle of List
  4. End of List
  5. Last remaining node in the list

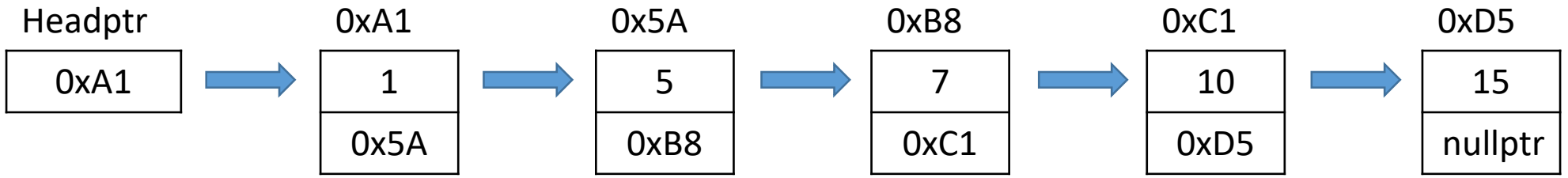
Empty List Removal

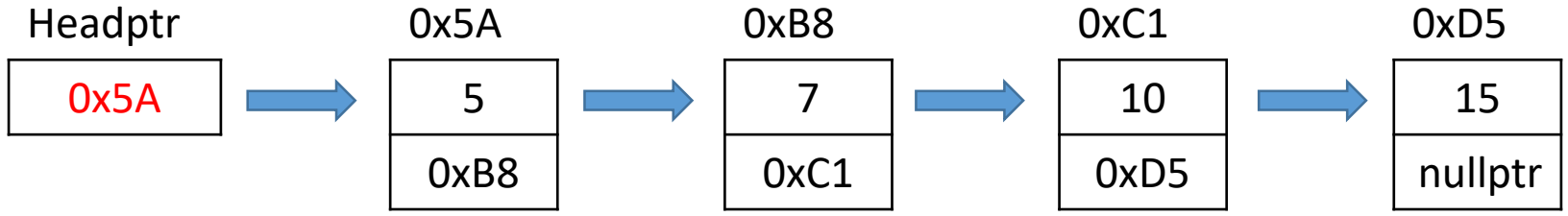
- Removal will fail but your routine should handle it



Front of List Removal

- Headptr will point to the remaining nodes in the list.

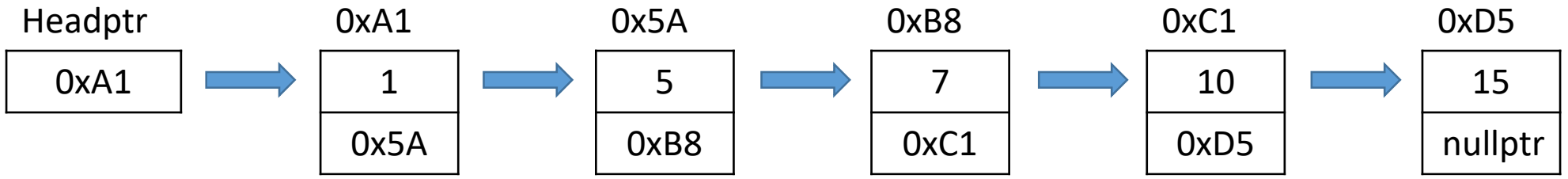




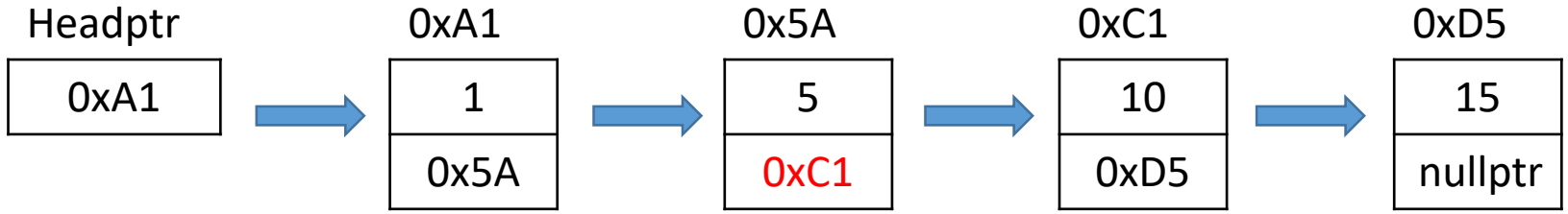
SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Middle of List Removal

- Headptr will point to the remaining nodes in the list.



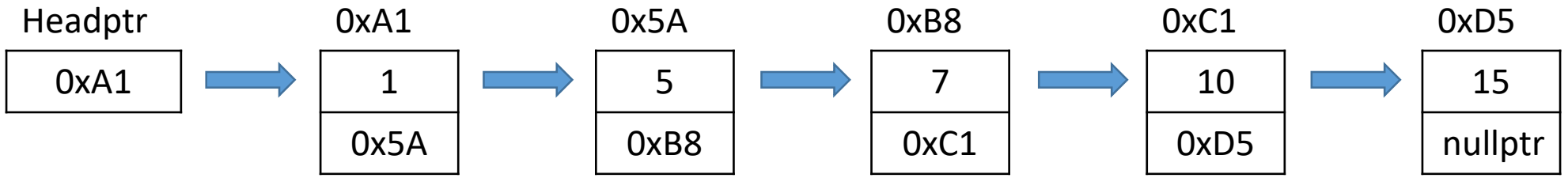
- Node before must bypass the node being removed.



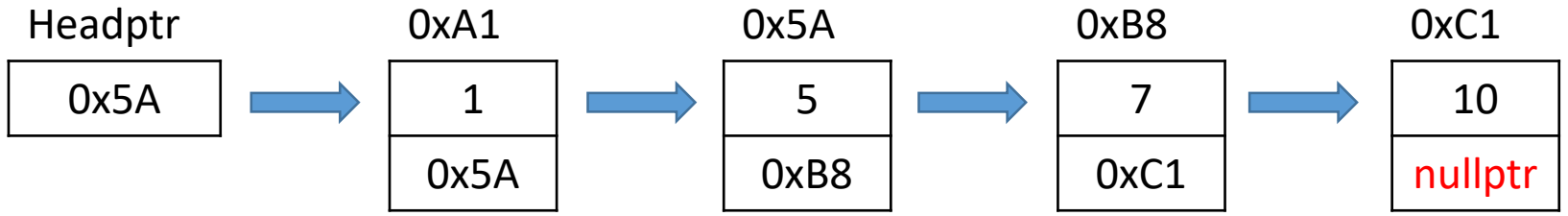
SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

End of List Removal

- Headptr will point to the remaining nodes in the list.



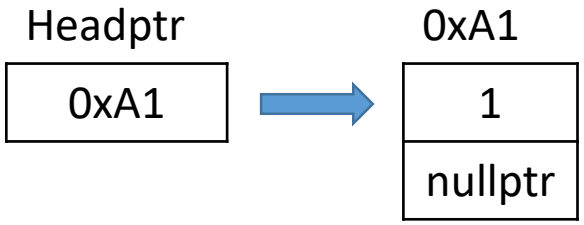
- Node just before the end must contain the nullptr.



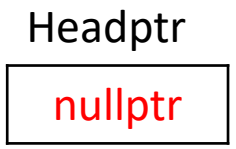
SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Last Node in the List Removal

- Is a remove from the front, but it is the only node in list



- headptr must contain the value of nullptr after removal



SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY