Singly Linked List using Unique pointer

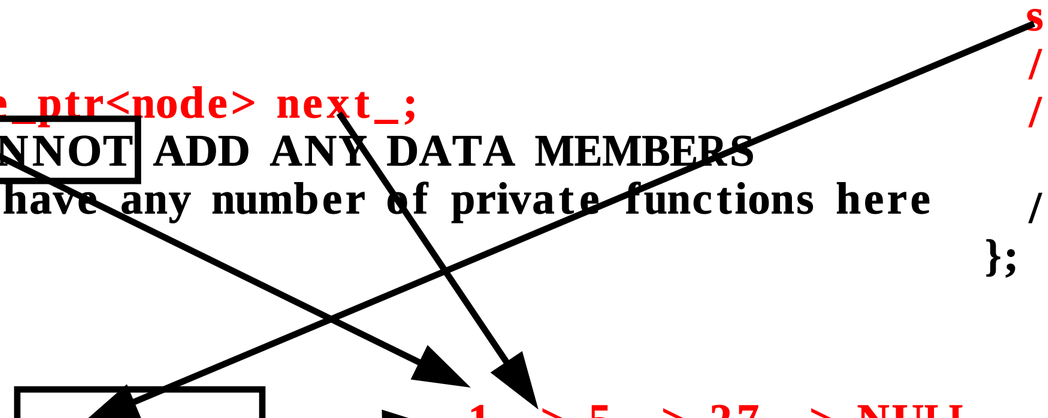
```
class node {
public:
  //Write all public functions below


private:
  T  data_;
  std::unique_ptr<node> next_;
  //YOU CANNOT ADD ANY DATA MEMBERS
  //You can have any number of private functions here
};
```

```
class slist1 {
public:
  //Write all public functions below

private:
  std::unique_ptr<node> first_;
  //YOU CAN ADD ANY OTHER DATA MEMBERS
  //For full grade both append and prepend MUST be O(1)

  //You can have any number of private functions here
};
```



You can have private

MUST USE ONLY std::unique_ptr for memory allocation

Must handle extremelly large list

slist1 s ;

1. s.append(T n) (Must be THETA(1))
2. s.prepend(T n) (Must be THETA(1))
3. cout 1->2->3->4->5->6->7->Null
4. s[i] if i < 0 or i >list size return NULL
5. bool find(T n)
6. bool remove(T n) if cannot remove, return false
7. slist1 s1(s) and s1 = s2;
8. (s1 == s2) and (s1 != s2)
9. Conversion function. if (s)
10. int n = size(s)