



Informatik

Übungsstunde 12

Prüfungsaufgabe 1

```
1 int* a = new int[6] {1, 3, 5, 7, 9, 11};  
2 int* x = a;  
3 x++;  
4 (*x)++;  
5 std::cout << *(a+*x);
```

Welche Aussage beschreibt die Ausgabe am besten? / Which statement describes the output best?

☐ 8



☐ 7



☐ 11



☐ 9



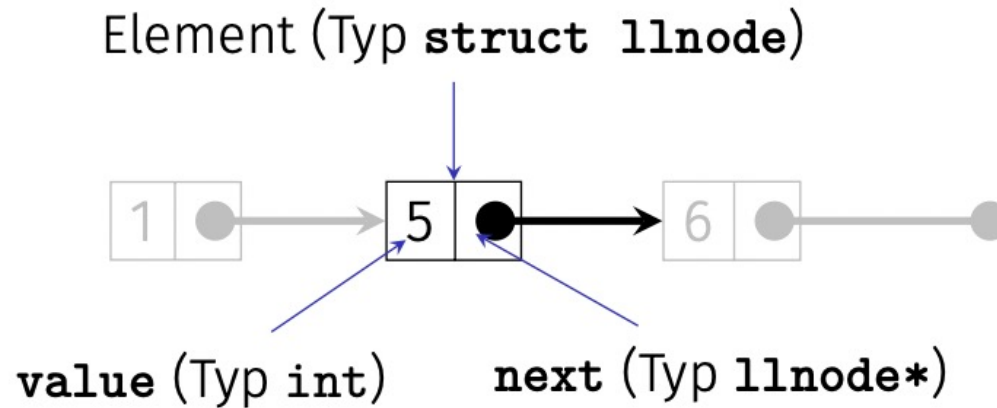
Prüfungsaufgabe 2

```
1 std::vector<int> v = {1, 2, 3};  
2 for (std::vector<int>::iterator it = v.begin();  
3     it != v.end();  
4     ++it) {  
5  
6     *it = 2 * *it;  
7 }
```

Welche Aussage passt am besten zum Code? / Which statement matches the code best?

- ☐ Kompiliert nicht / Doesn't compile \$
- ☐ Der Code erzeugt einen Speicherzugriffsfehler / The code produces a memory access error \$
- ☐ Nach der Schleife enthält v die Werte 1 2 3 / After the loop, v contains the values 1 2 3 \$
- ☐ Nach der Schleife enthält v die Werte 2 4 6 / After the loop, v contains the values 2 4 6 \$

Zusammenfassung



```
struct llnode {  
    int value;  
    llnode* next;  
  
    llnode(int v, llnode* n): value(v), next(n) {} // Constructor  
};
```

Zusammenfassung

```
class llvec {  
    llnode* head;  
public: // Public interface identical to avec's  
    llvec(unsigned int size);  
    unsigned int size() const;  
    ...  
};
```

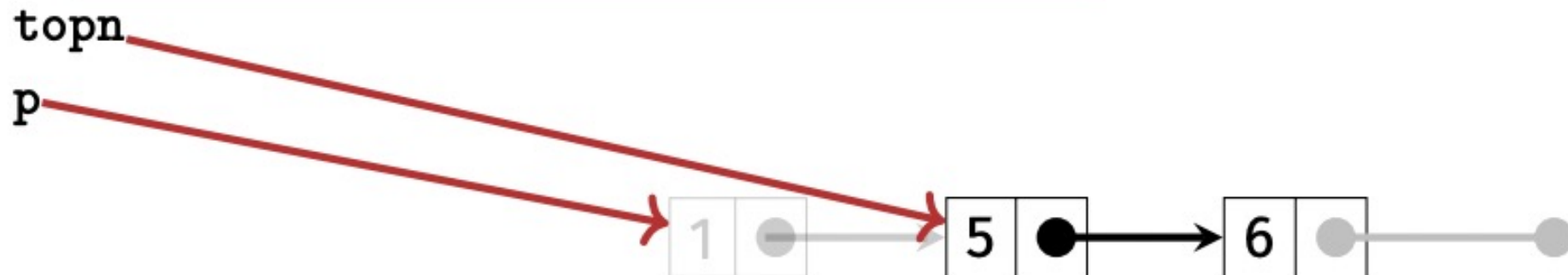
Zusammenfassung

```
void llvec::push_back(int e) {  
    if (this->head == nullptr) { // Case 1: empty list  
        this->head = new llnode{e, nullptr};  
    } else { // Case 2: non-empty list (code as before)  
        llnode* n = this->head;  
        for (; n->next != nullptr; n = n->next);  
  
        n->next = new llnode{e, nullptr};  
    }  
}
```

delete

```
void stack::pop(){  
    assert (!empty());  
    llnode* p = topn;  
    topn = topn->next;  
    delete p;  
}
```

Erinnerung: Abkürzung für `(*topn).next`



Destruktor

```
// POST: the dynamic memory of *this is deleted
stack::~~stack(){
    while (topn != nullptr){
        llnode* t = topn;
        topn = t->next;
        delete t;
    }
}
```