Exercise Week 06

GianAndrea Müller
mailto:muellegi@student.ethz

April 22, 2018

Exercise Week 06

GianAndrea Müller
mailto:muellegi@student.ethz

April 22, 2018

Time Schedule
- 15' Bubble sort & Maximum sort
- 15' 2D Arrays
- 5' Matric as Array
- 15' Strings
- 15' Pointers
- 5' Arrays und Pointers

Time Schedule

- 15' Bubble sort & Maximum sort
- 15' 2D Arrays
- 5' Matrix as Array
- 15' Strings
- 15' Pointers
- 5' Arrays und Pointers

Learning Objectives
Verständnis des Konzepts von Pointern
Kenntnis der Funktionalität von Strings

Learning Objectives

- Verständnis des Konzepts von Pointern
- Kenntnis der Funktionalität von Strings

Bubble sort
Bubble sort slides

Bubble sort

Bubble sort slides

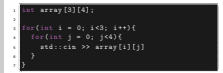
Max sort
Max sort slides

Max sort

- Idea: Improve on bubble sort by not swapping every time but searching for the max and swapping when having found it.

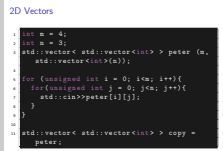
Max sort slides

2D Arrays



2D Arrays

```
1 int array[3][4];
2
3 for(int i = 0; i<3; i++){
4   for(int j = 0; j<4){
5     std::cin >> array[i][j]
6   }
7 }
```



- Bei der Initialisierung eines 2D Vektors wird ein Vektor von Vektoren angelegt.

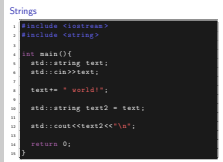
2D Vectors

```
1  int m = 4;
2  int n = 3;
3  std::vector< std::vector<int> > peter (m,
       std::vector<int>(n));
4
5  for (unsigned int i = 0; i<m; i++){
6    for(unsigned int j = 0; j<n; j++){
7      std::cin>>peter[i][j];
8    }
9  }
10
11 std::vector< std::vector<int> > copy =
       peter;
```

Matrix as Array
Matrix as Array slides

Matrix as Array

Matrix as Array slides

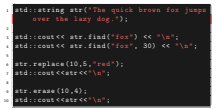


- Die Bibliothek string erlaubt das Speichern von Buchstaben in einer Vektor-aehnlichen Struktur.
- Sie hat wie der Vektor Zugriffsmethoden wie .at() und .push_back().
- Zusaetzlich kann ueber den +=-Operator ein string einfach angehaengt werden.
- Weiter koennen cin und cout direkt verwendet werden. Fuer das Einlesen und Ausgeben von mehreren Buchstaben wird keine Schleife benoetigt.

Strings

```
1  #include <iostream>
2  #include <string>
3
4  int main(){
5    std::string text;
6    std::cin>>text;
7
8    text+= " world!";
9
10   std::string text2 = text;
11
12   std::cout<<text2<<"\n";
13
14   return 0;
15 }
```

String



String

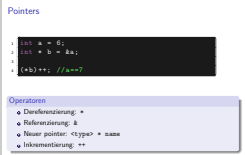
- Die .find() Funktion erlaubt es einen Substring innerhalb des Strings zu lokalisieren. Rueckgabewert ist dabei die Position des ersten Buchstabens des Substrings.
- Weiter kann als zweites Argument die Startposition der Suche uebergeben werden. Das ist nuetzlich wenn der gesuchte Substring mehrmals vorkommen kann.
- Falls der Substring nicht gefunden wird, wird die Konstante std::string:npos zurueckgegeben (eine grosse positive Zahl, Wert compilerabhaengig).

```
1  std::string str("The quick brown fox jumps
       over the lazy dog.");
2
3  std::cout<< str.find("fox") << "\n";
4  std::cout<< str.find("fox", 30) << "\n";
5
6  str.replace(10,5,"red");
7  std::cout<<str<<"\n";
8
9  str.erase(10,4);
10 std::cout<<str<<"\n";
```

Pointers

Pointers

```
1 int a = 6;
2 int & b = a;
3
4 b++; //a==7
```



- Der Dereferenzierungsoperator wird gebraucht um den Wert auf den ein Pointer zeigt zu erhalten. Er wird immer vor den Pointer gesetzt.
- Der Referenzierungsoperatur wird gebraucht um die Adresse einer Variable zu erhalten.
- Um einen neuen Pointer zu erzeugen wird in der Variablendeklaration nach dem Typ der Variable ein Stern angehaengt.
- Der inkrementierungsoperator verschiebt den Pointer um die Laenge des Datentyps auf den er zeigt.

Pointers

```
1 int a = 6;
2 int * b = &a;
3
4 (*b)++; //a==7
```

Operatoren

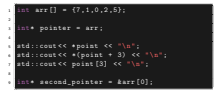
- Dereferenzierung: `*`
- Referenzierung: `&`
- Neuer pointer: `<type> * name`
- Inkrementierung: `++`

Einführung zu Pointern
Einführung zu Pointern slides

Einführung zu Pointern

Einführung zu Pointern slides

Pointers



Pointers

```
1 int arr[] = {7,1,0,2,5};
2
3 int* pointer = arr;
4
5 std::cout<< *point << "\n";
6 std::cout<< *(point + 3) << "\n";
7 std::cout<< point[3] << "\n";
8
9 int* second_pointer = &arr[0];
```

Arrays und Pointer
Arrays und Pointer slides

Arrays und Pointer

Arrays und Pointer slides