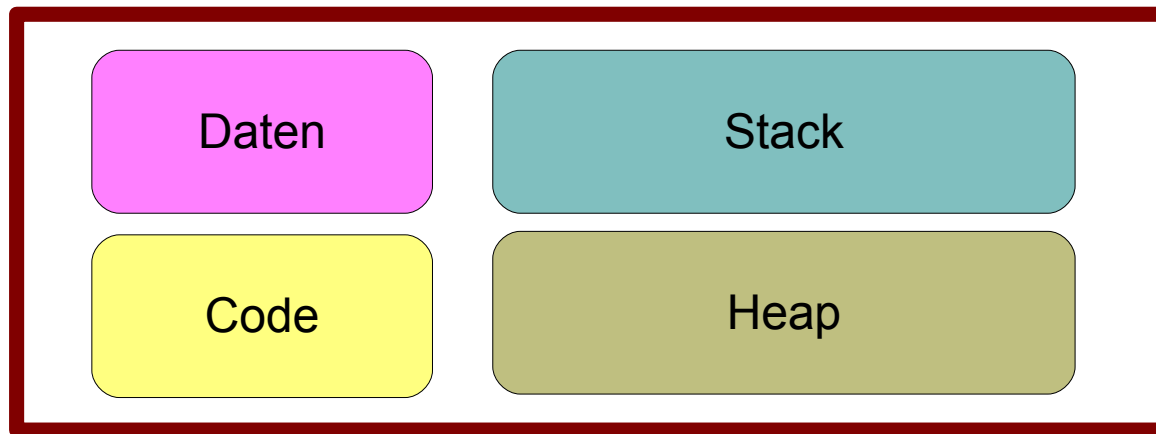
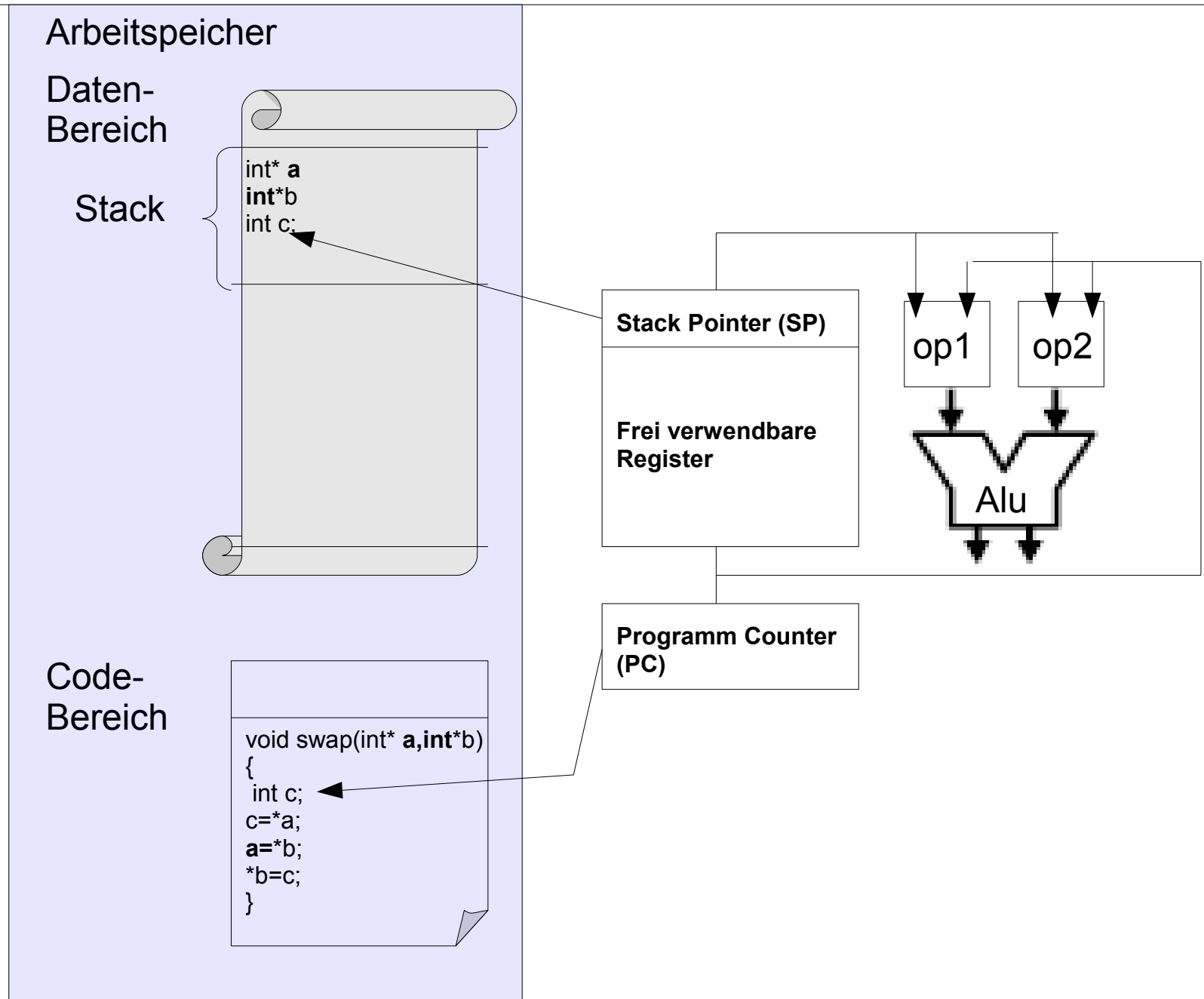

Vorlesung8

- Speicherbereiche: Parametermechanismus.
 - Stack Pointer
 - Lokale Variablen und Frei verwendbare Register
 - Daten Bereiche und Code Bereiche.
- Sortieralgorithmen
 - Auswahlort (direktes Auswählen)
 - Bubblesort.

Der Speicher

▣ *Speicherbereiche in laufenden Programmen*





Speicherbereiche

- CODE:** Hier werden die Befehle des Programms geladen und daraus in den Prozessor übermittelt.
- DATEN:** In diesem Speicherbereich bleiben die Daten, die verfügbar bis zum Ende des Programms sind (***globale und statische Variablen***)
- STACK (Stapel):** Behält die Funktionsaufrufe und ***lokalen Variablen***. (Bis zum 8 Megabytes)
- HEAP:** Hier wird **dynamischen Speicherplatz** reserviert. (theoretisch bis zum 4 Gigabytes)

Globale Variable

int c; // **Globale Deklaration: Außerhalb des mains**

void summe(void); /*Kopf der Funktion (Deklaration)*/

int main(void)

{

short a, b;

/* c wird nicht im main deklariert */

a = 3000; b = 5200; /* Initial.*/

summe(); /* Aufruf */

cout << c; /* Ausgabe auf dem Bildschirm */

return 0;

}

/*Körper der Funktion (Definition)*/

void summe (void)

{ short a,b;

/* c wird nicht in der Funktion deklariert */

c = a + b;

return;

}

Der Stack: Bereiche

- Lokalen Variablen Verarbeitung.
- Aufruf der Funktionen

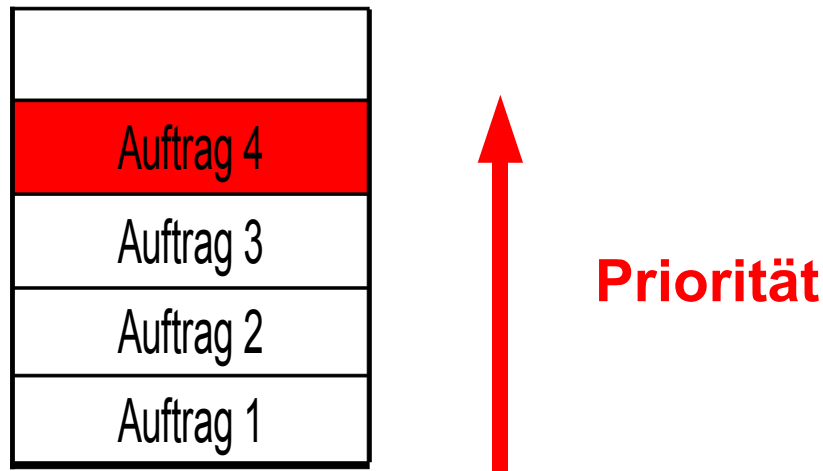
Der Stack (Der Stapel)

Ein wichtiger Speicherbereich

STACK (Stapel): Beinhaltet die Funktionsaufrufe und **lokalen Variablen**.

Dessen Funktionsweise kann uns dabei helfen:

- Die **Aufrufe** einer Funktion besser zu verstehen.
- Warum die Variablen **Gültigkeitsbereich** haben?



Der Stack : LIFO

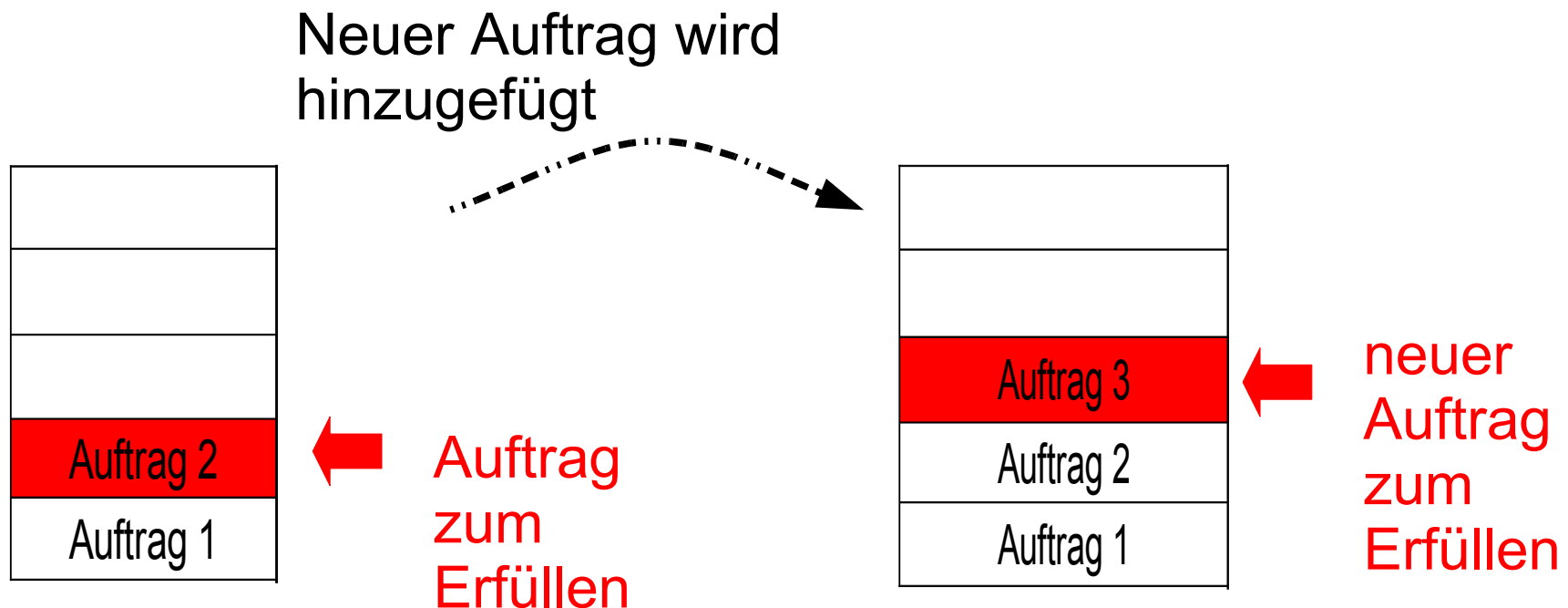
Die letzten Aufträge, die auf den Stapel gelegt wurden, sind die ersten, die erfüllt werden müssen.

LIFO: Last In First Out



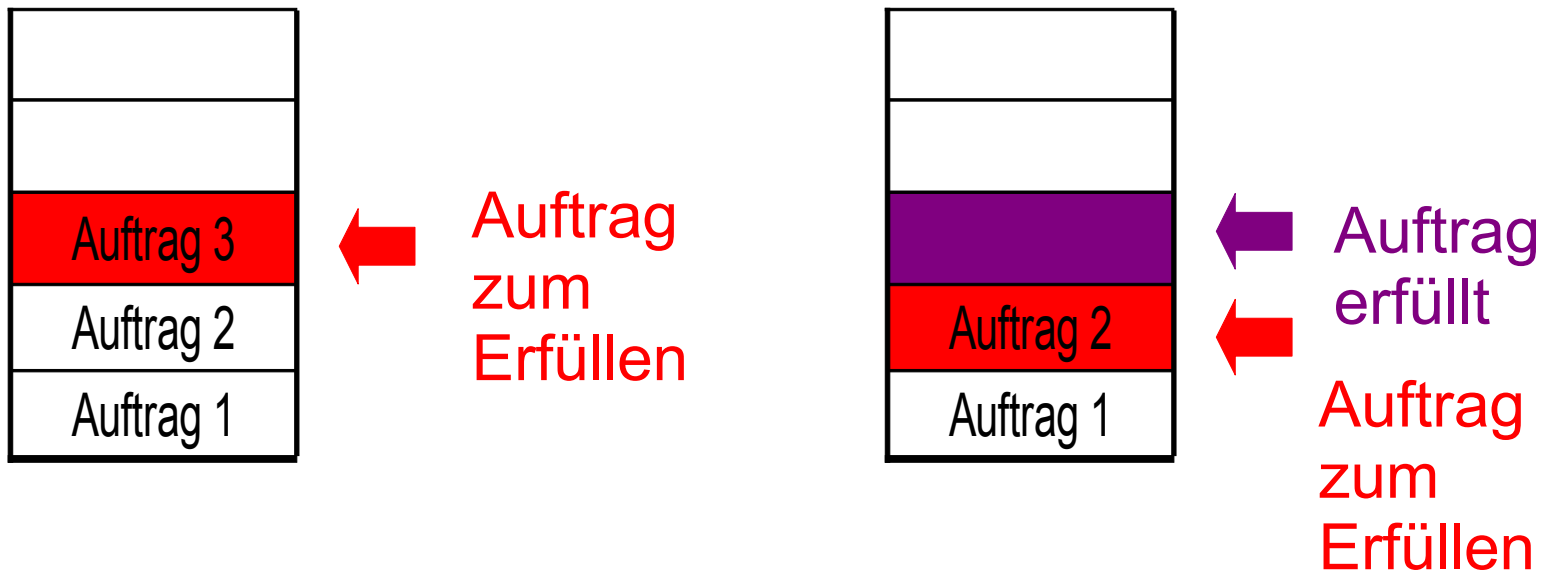
Neuer Auftrag: stapeln

- Der Stack enthält verschiedene gestapelte Aufträge, die müssen in dieser **Reihenfolge** erfüllt werden.
- Neue Aufträge zum Erfüllen werden immer auf dem Stapel gelegt.



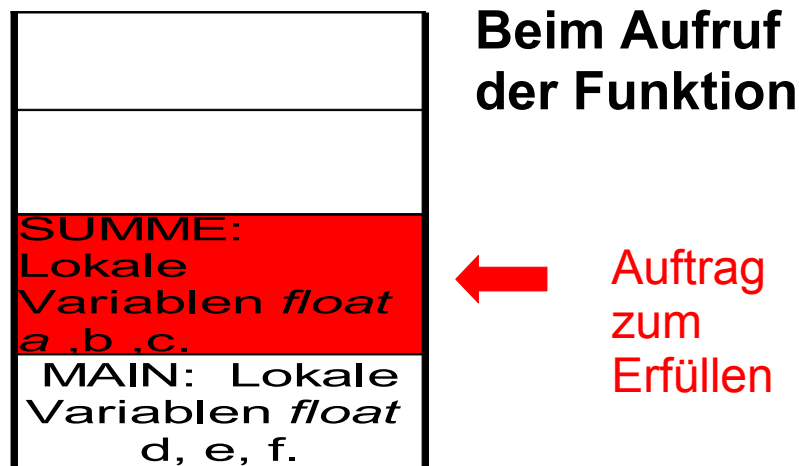
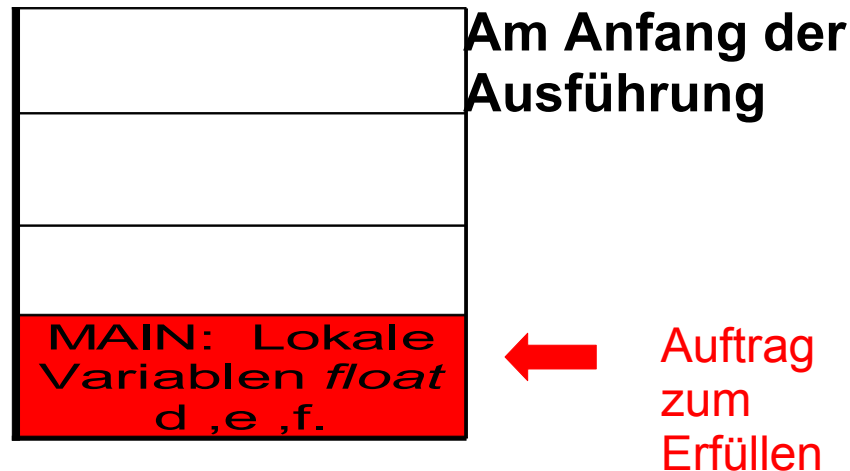
Auftrag erledigen: entstapeln

- Wenn der Auftrag **erfüllt** ist, wird er aus dem Stapel **entfernt**.



Der Stack: Funktionsaufruf

Praktische Anwendung im laufenden Programm



Algorithmen: Sortierung

- Aufgabe:

Die Folgenden Zahlen sollen sortiert werden:

Am Anfang: 13 6 27 9 1

Am Ende: 1 6 9 13 27

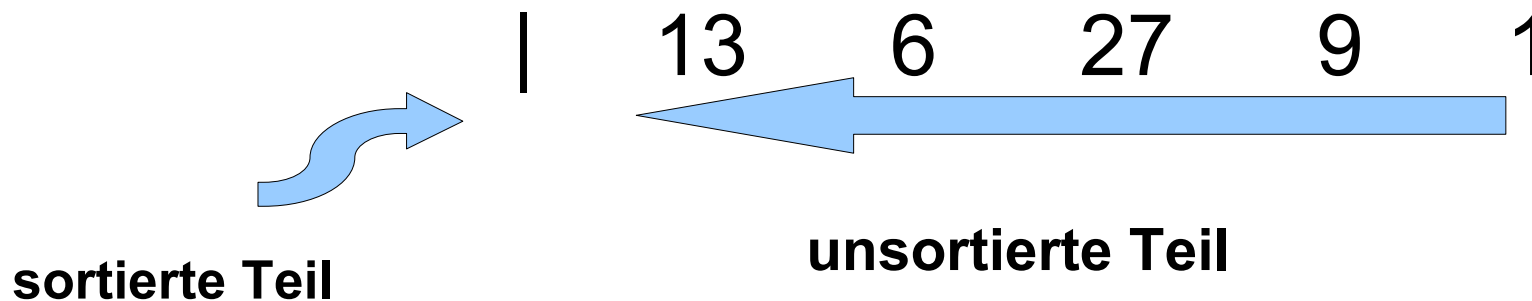
Lösungsweg(II)

Auswahlort (direktes Auswählen)

Algorithmen: Sortierung

Lösungsweg II

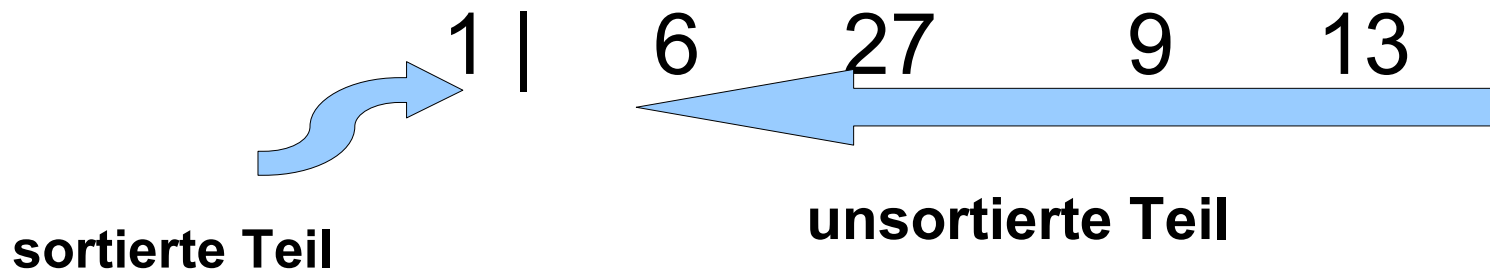
- Sortieren durch direktes Auswählen:



Algorithmen: Sortierung

Lösungsweg II

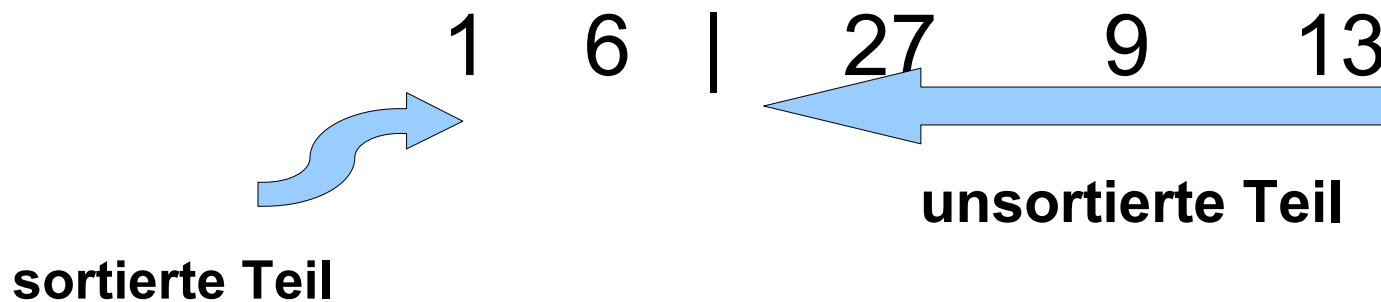
- Sortieren durch direktes Auswählen:



Algorithmen: Sortierung

Lösungsweg II

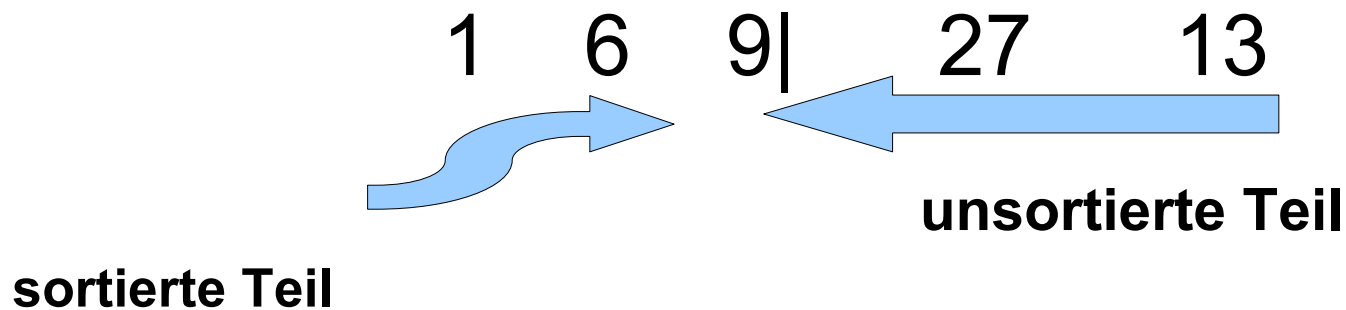
- Sortieren durch direktes Auswählen:



Algorithmen: Sortierung

Lösungsweg II

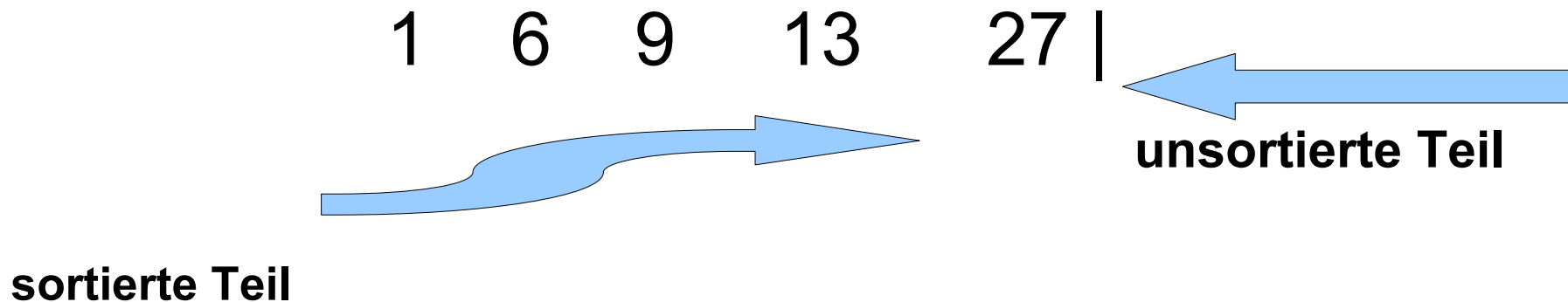
- Sortieren durch direktes Auswählen:



Algorithmen: Sortierung

Lösungsweg II

- Sortieren durch direktes Auswählen:



Algorithmen: Sortierung

- Sortieren durch direktes Auswählen:

Der Vorteil von Sortieren durch Auswählen gegenüber Sortieren durch Einfügen liegt darin ,dass pro Element zwar mehrere Vergleiche jedoch nur eine Vertauschung stattfindet.

Dies spielt eine Rolle, wenn große Datensätze sortiert werden,da dann im allgemeinen die Vergleichsoperation schneller als das vertauschen ist.

Vorlesung8_1.cpp

Algorithmen: Sortierung

Aufgabe2:

Es soll ein Programm geschrieben werden, das die folgenden Zahlen-Arrays in aufsteigender Reihenfolge sortiert:

99 13 45 12 17 33 67 1

Algorithmen: Sortierung

Lösungsweg:

Beschreibung des Verfahren:

- Das zu sortierende Array mehrmals von rechts nach links durchlaufen.
- Das jeweilige Element mit seinem linken vertauschen, falls die beiden nicht in der richtigen Reihenfolge stehen.
- Die kleineren Elemente wandern also von rechts nach links.

Algorithmen: Sortierung

Lösungsweg:

Beschreibung des Verfahren:

	99	13	45	12	17	33	67	1
1		99	13	45	12	17	33	67

Algorithmen: Sortierung

Lösungsweg:

Beschreibung des Verfahren:

		99	13	45	12	17	33	67	1
1			99	13	45	12	17	33	67
1	12			99	13	45	17	33	67

Algorithmen: Sortierung

Lösungsweg:

Beschreibung des Verfahren:

		99	13	45	12	17	33	67	1
1			99	13	45	12	17	33	67
1	12			99	13	45	17	33	67
1	12	13			99	17	45	33	67

Algorithmen: Sortierung

Lösungsweg:

Beschreibung des Verfahren:

		99	13	45	12	17	33	67	1
1			99	13	45	12	17	33	67
1	12			99	13	45	17	33	67
1	12	13			99	17	45	33	67
1	12	13	17			99	33	45	67

Algorithmen: Sortierung

Lösungsweg:

Beschreibung des Verfahren:

		99	13	45	12	17	33	67	1
1			99	13	45	12	17	33	67
1	12			99	13	45	17	33	67
1	12	13			99	17	45	33	67
1	12	13	17			99	33	45	67
1	12	13	17	33			99	45	67

Algorithmen: Sortierung

Lösungsweg:

Beschreibung des Verfahren:

		99	13	45	12	17	33	67	1
1			99	13	45	12	17	33	67
1	12			99	13	45	17	33	67
1	12	13			99	17	45	33	67
1	12	13	17			99	33	45	67
1	12	13	17	33			99	45	67
1	12	13	17	33	45			99	67

Algorithmen: Sortierung

Lösungsweg:

Beschreibung des Verfahren:

		99	13	45	12	17	33	67	1
1			99	13	45	12	17	33	67
1	12			99	13	45	17	33	67
1	12	13			99	17	45	33	67
1	12	13	17			99	33	45	67
1	12	13	17	33			99	45	67
1	12	13	17	33	45			99	67
1	12	13	17	33	45	67			99

Algorithmen: Sortierung

Lösungsweg:

Beschreibung des Verfahren:

Das Verfahren, das wir benutzt haben, heißt „Bubble-sort“: Stellt man sich das Feld senkrecht und die kleineren Elemente als 'leichtere' Blasen (bubbles) vor, die im Feld in eine ihrem Gewicht entsprechende Stelle aufsteigen, so findet man die Erklärung für den Namen Bubble -Sort.

Algorithmen: Sortierung

Lösungsweg:

Beschreibung des Verfahren:

Für Bubble-Sort gibt es verschiedene Varianten. Sie unterscheiden sich im wesentlichen im laufbereich des Indizes.

vorlesung8_2.cpp

vorlesung8_3.cpp