WBS TRAINING

Programmierung(2)

UNSER ♥ SCHLÄGT FÜR BILDUNG

Agenda

- String (Zeichenkette)
  - **Definition, Motivation** und Beispiel
  - Darstellung im **PAP**
  - Darstellung im **Struktogramm**
  - Darstellung im **Pseudocode**
  - Syntax in **ANSI C**
    - **‘\0‘** (Terminator)
    - **Pointer** (auf einen String/Character-Array)
    - **Eingabe** (mit scanf)
      - String ohne Leerzeichen
      - String mit Leerzeichen
    - **Ausgabe** (mit printf)
- Stringfunktionen
  - **strcpy**
  - **strcmp**
  - **strlen**
  - **strcat**
- Fachpraktische Anwendungen

String (Zeichenkette) – **Definition** und **Motivation**

- Ein **String** ist eine Folge von Charactern mit einem eindeutig definierten Endzeichen.
- Strings erlauben uns nun also, auch mit Wörtern, Sätzen oder ganzen **Texten** zu arbeiten.
- Da es für Strings in der Programmierung **kein graphisches Symbol** gibt, werden wir diese im PAP, Struktogramm und Pseudocode in der Regel nur sprachlich notieren.
- Da Strings in ANSI C **keinen eigenständigen Typen** darstellen (im Unterschied zu Integer, Float, Character …), werden wir auf einige gewohnte Techniken verzichten und Alternativen kennen lernen müssen.
- Dennoch können wir Strings veranschaulichen, indem wir uns diese als ***„Text-Variablen***“ vorstellen.
- Entsprechend werden wir uns damit beschäftigen, wie man …
  - Strings deklariert, definiert und initialisiert
  - Strings (vom User) abfragt
  - Strings (auf der Konsole) ausgibt

String (Zeichenkette) – **Beispielaufgabe**

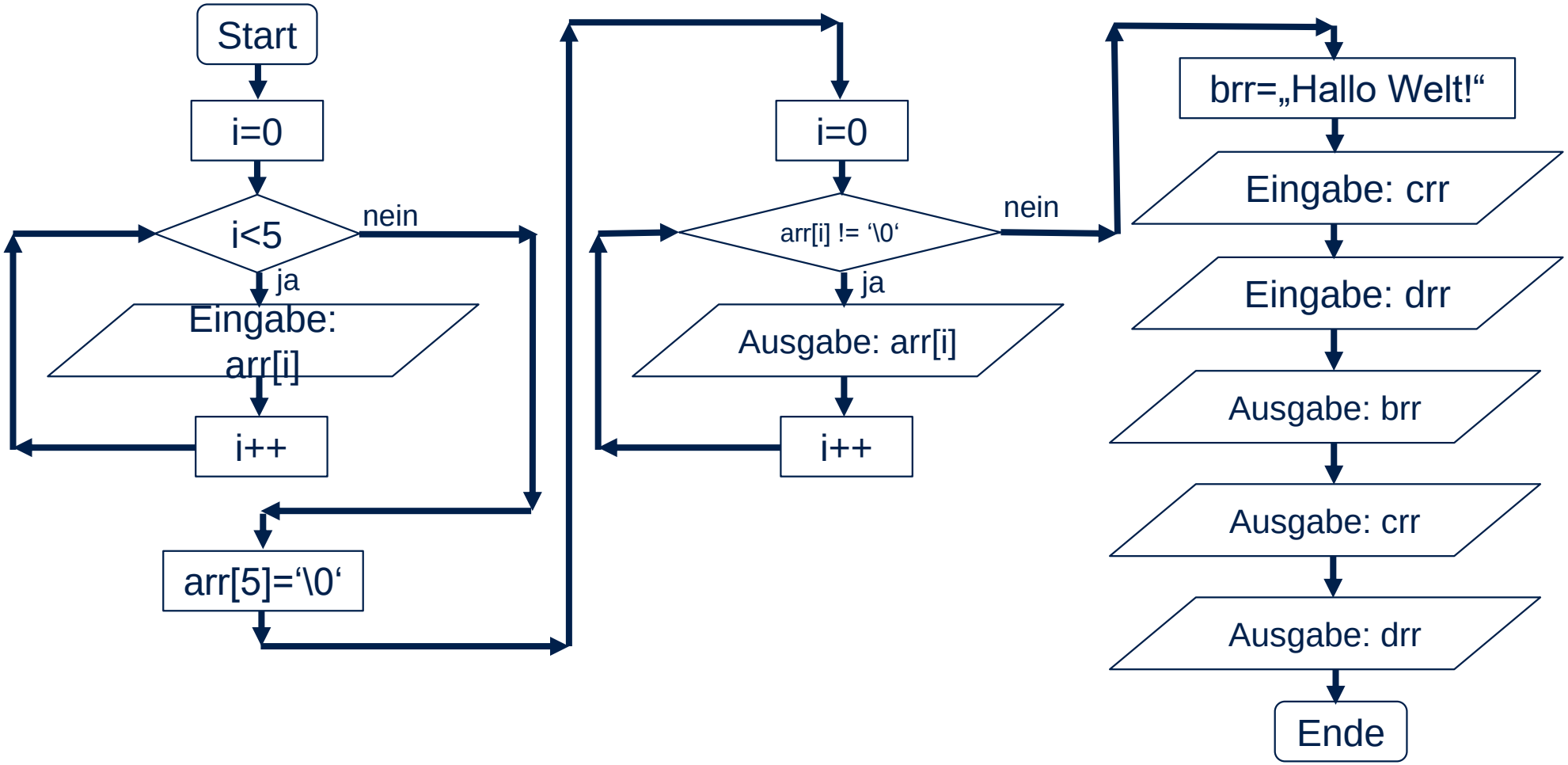
Aufgabenstellung

Vorbemerkung: Die folgende Aufgabe erhebt nicht den Anspruch, inhaltlich interessant zu sein. Vielmehr geht es darum, in kompakter Form die wichtigsten Techniken mit Strings kennenzulernen. Ferner soll ein Blick „hinter die Kulissen“ ermöglicht werden, der uns verwehrt bliebe, wenn wir von Beginn an ausschließlich mit „Stringfunktionen“ gearbeitet hätten.

- Das Programm fragt vom User zunächst mittels einer Schleife 5 Buchstaben ab.
- Die Buchstaben werden in arr[0] bis arr[4] abgespeichert. (arr ist ein 10-Felder-langes Character-Array)
- In arr[5] wird ‘\0‘ (“Terminator“) abgespeichert. (Dieses Zeichen symbolisiert, dass der String hier endet)
- Anschließend startet eine Ausgabe-Schleife, die alle Zeichen des Strings in arr (ausschließlich ‘\0‘) ausgibt.
- Nach der Schleife soll …
  a) das Character-Array brr[20] mit dem String „Hallo Welt!“ initialisiert werden
  b) das Character-Array crr[15] per User-Eingabe mit einem **Wort** (ohne Leerzeichen) gefüllt werden
  c) das Character-Array drr[50] per User-Eingabe mit einem **Satz** (mit Leerzeichen) gefüllt werden
- Abschließend werden die Strings aus brr, crr und drr auf der Konsole ausgegeben und das Programm endet.

Auch für diese Aufgabe wollen wir zunächst **PAP**, **Struktogramm** und **Pseudocode** erstellen, um erst daraufhin den entsprechenden **Quellcode** in ANSI C zu codieren.

String (Zeichenkette) – **Beispielaufgabe** – PAP

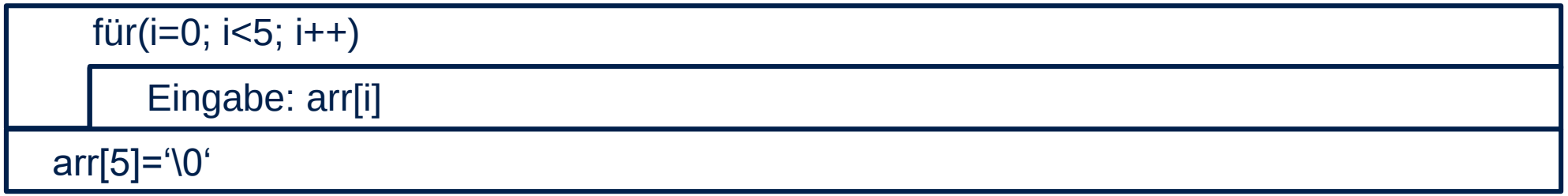


String (Zeichenkette) – **Beispielaufgabe** – **Struktogramm**

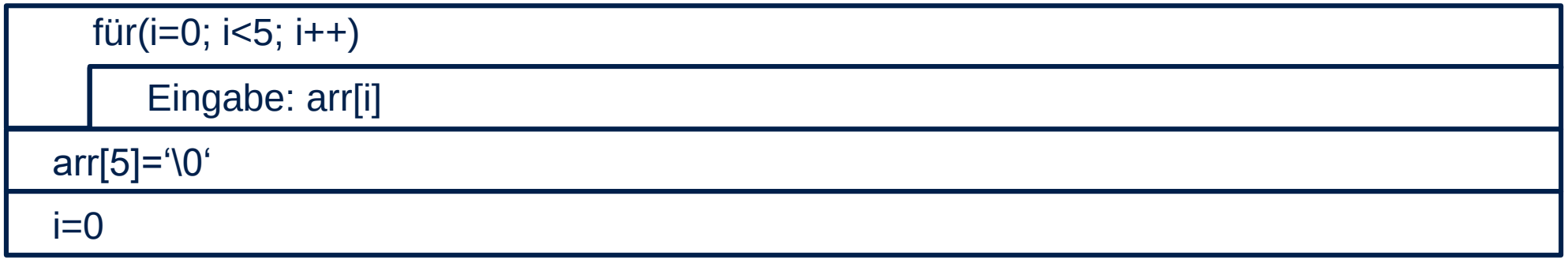
String (Zeichenkette) – **Beispielaufgabe** – **Struktogramm**

String (Zeichenkette) – **Beispielaufgabe** – **Struktogramm**

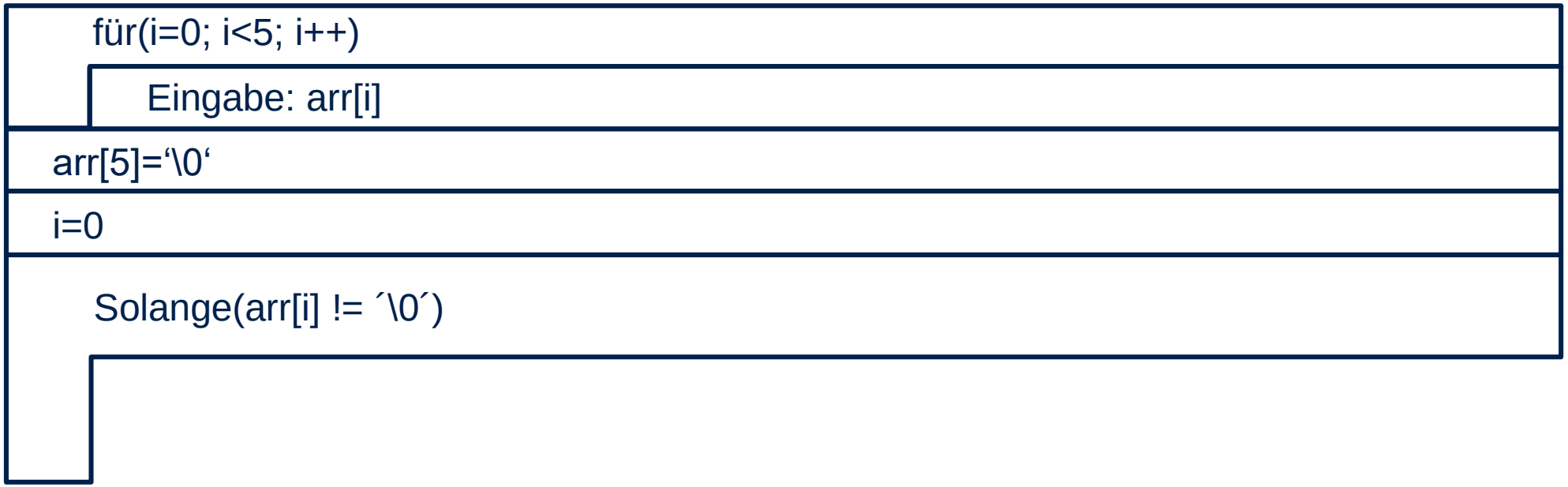
String (Zeichenkette) – **Beispielaufgabe** – **Struktogramm**



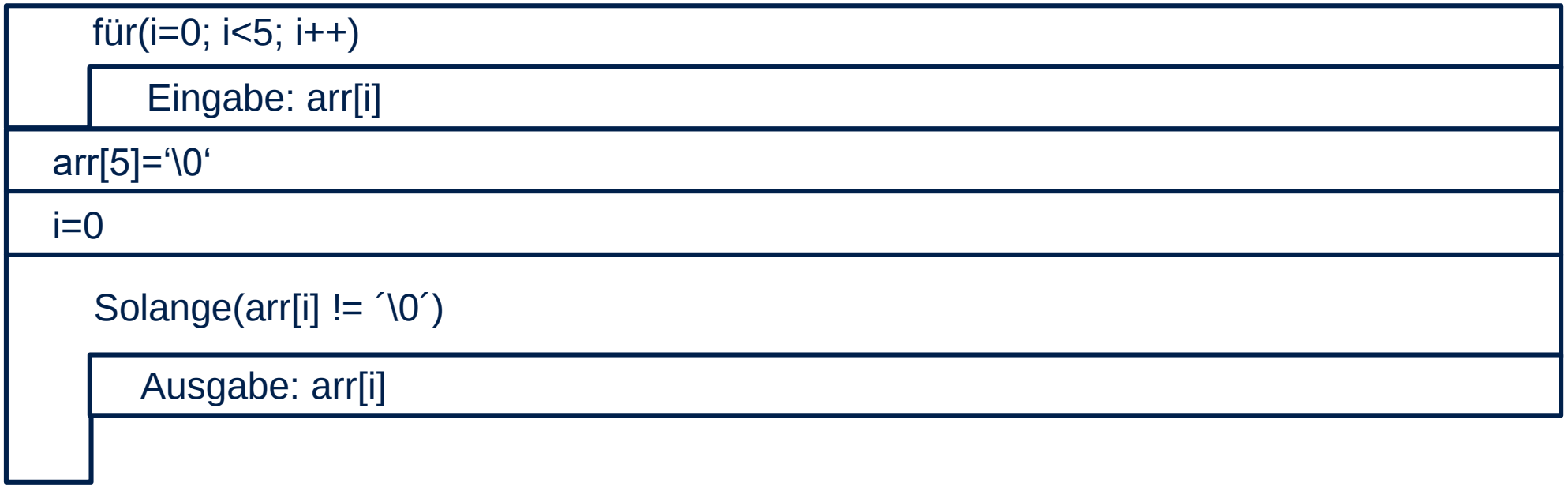
String (Zeichenkette) – **Beispielaufgabe** – **Struktogramm**



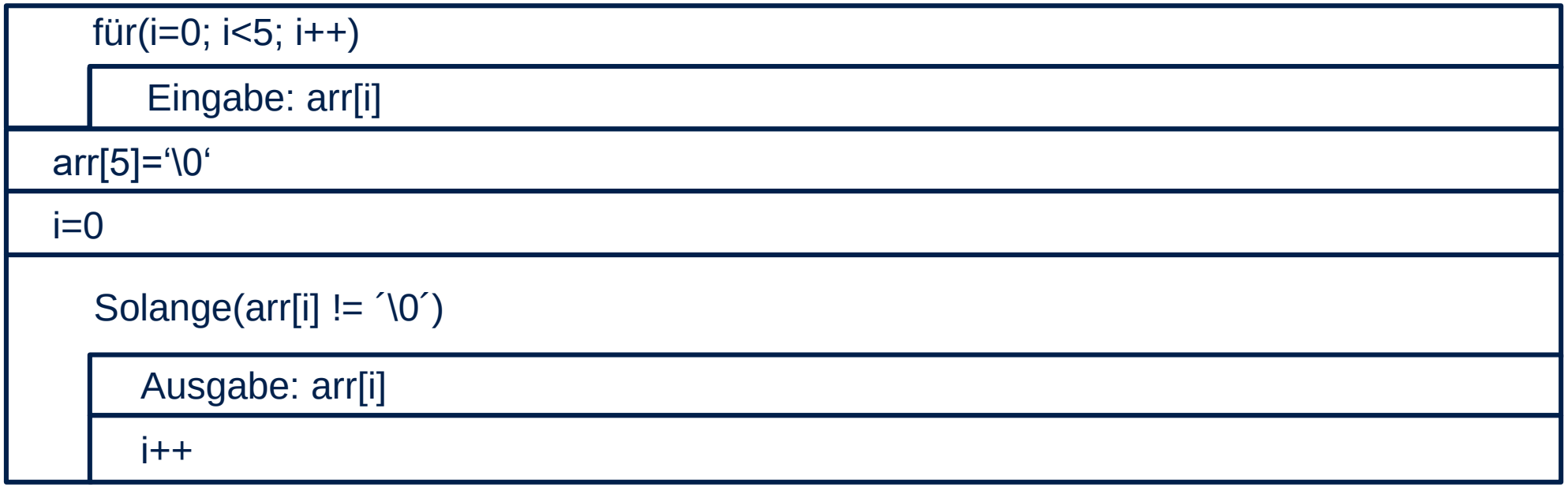
String (Zeichenkette) – **Beispielaufgabe** – **Struktogramm**



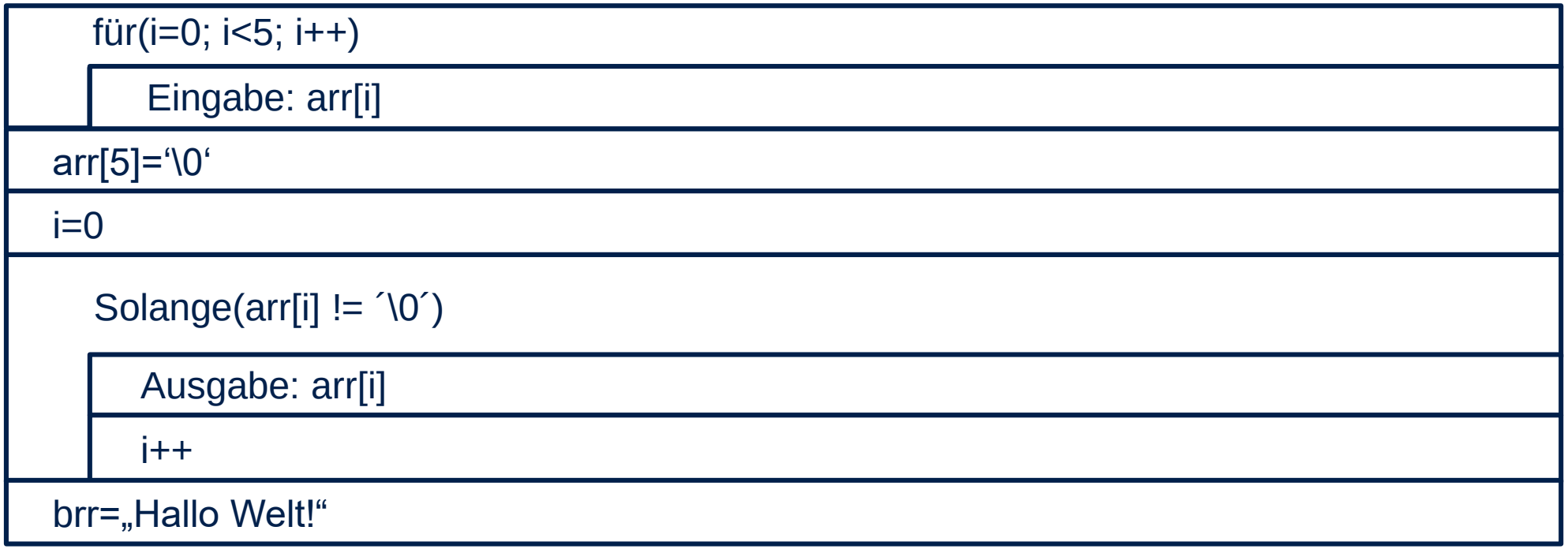
String (Zeichenkette) – **Beispielaufgabe** – **Struktogramm**



String (Zeichenkette) – **Beispielaufgabe** – **Struktogramm**

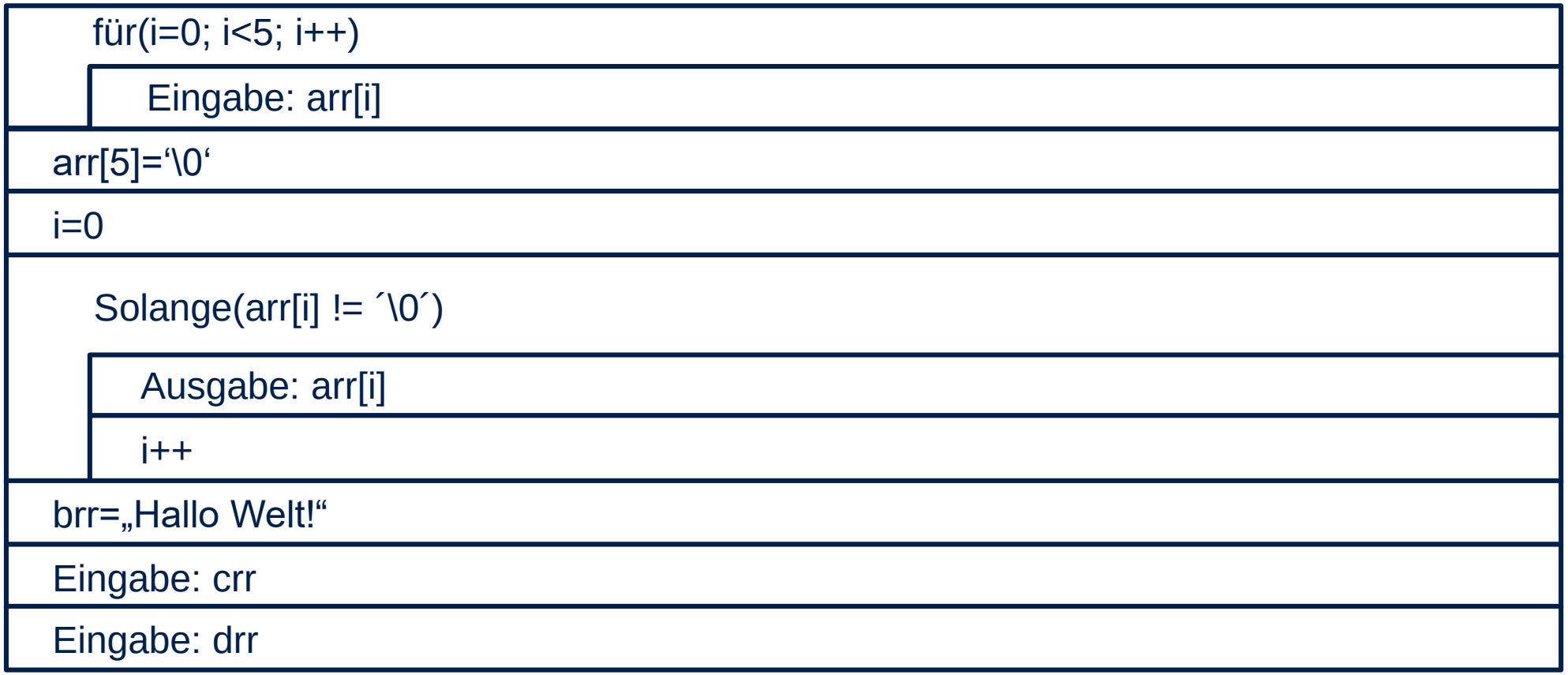


String (Zeichenkette) – **Beispielaufgabe** – **Struktogramm**

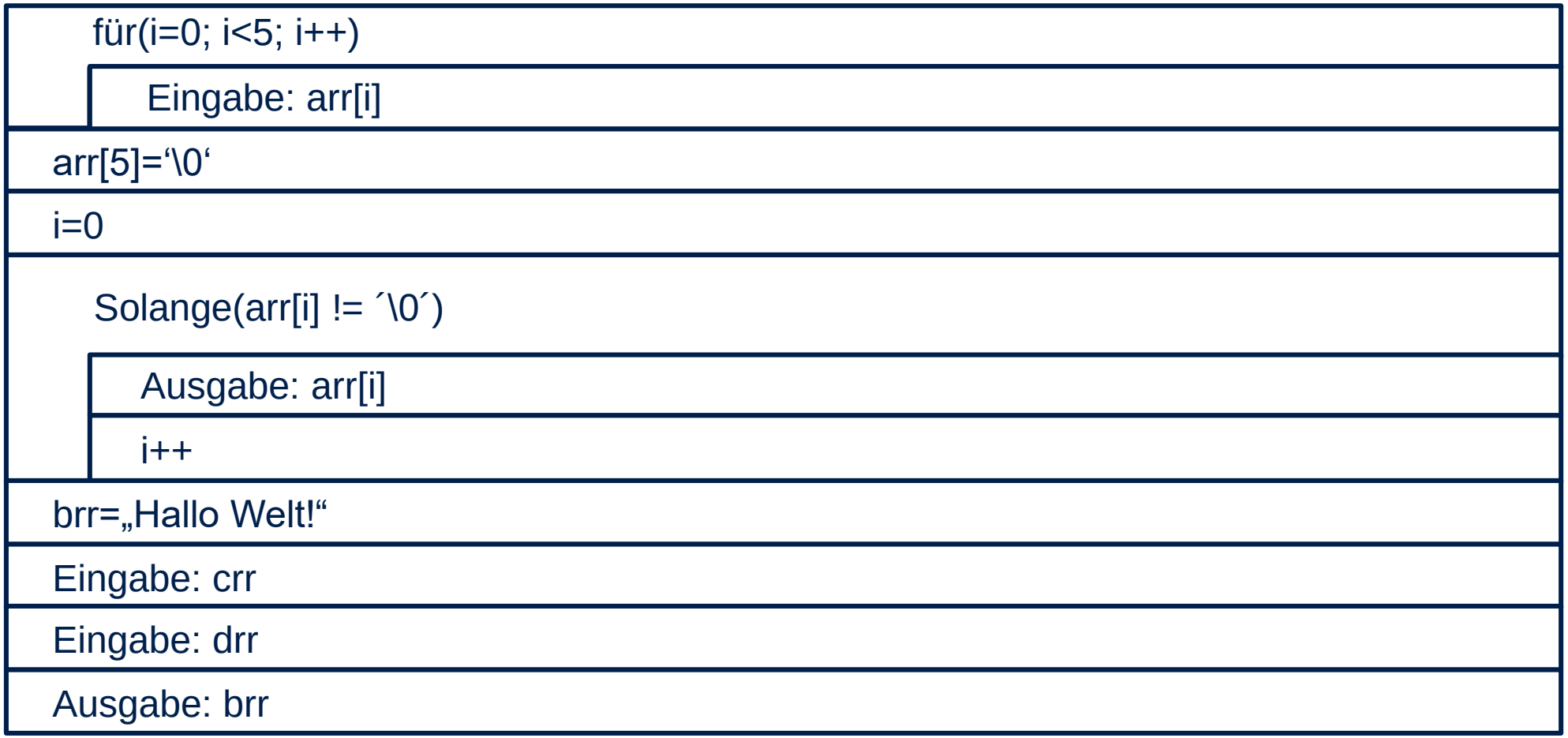


String (Zeichenkette) – **Beispielaufgabe** – **Struktogramm**

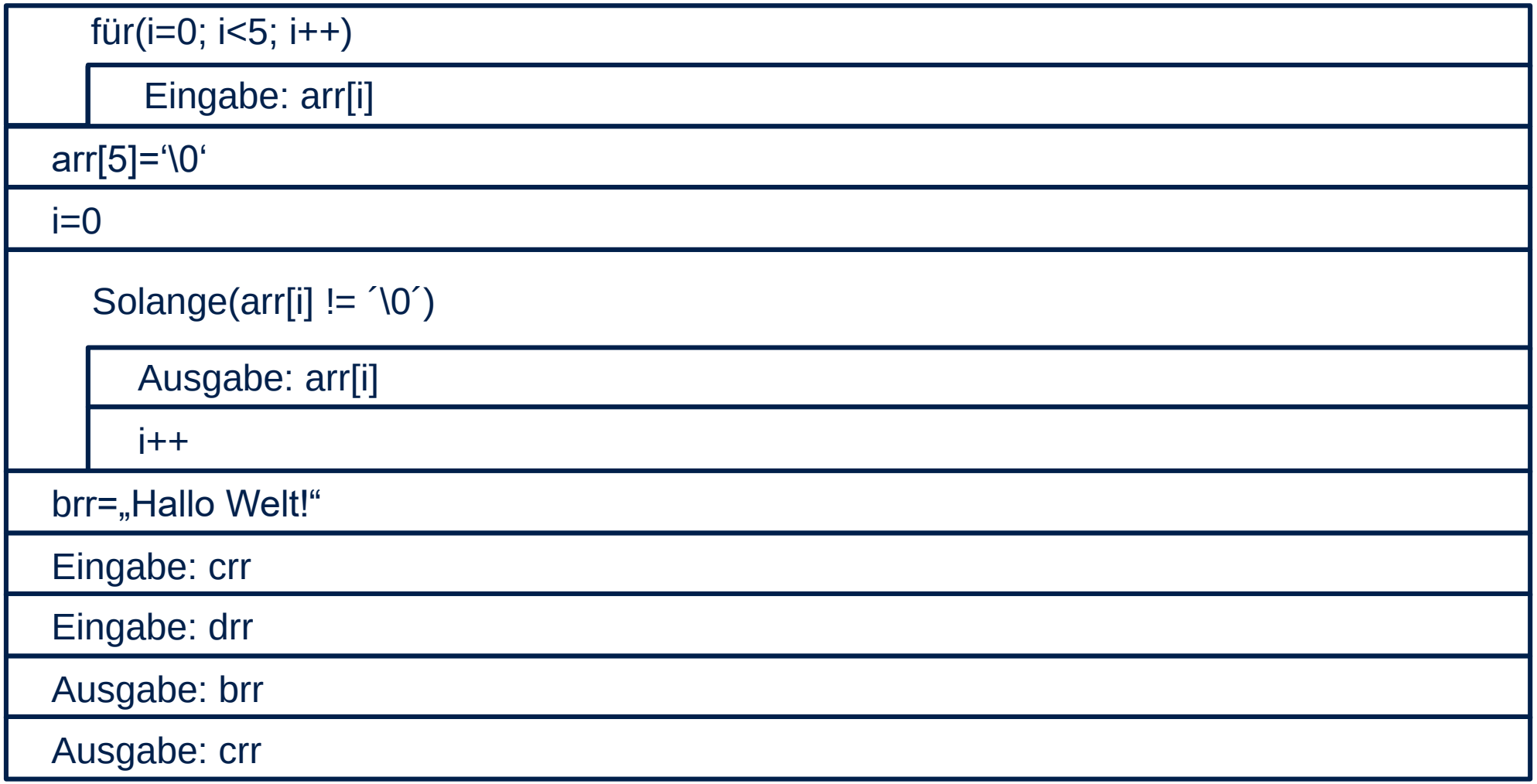
String (Zeichenkette) – **Beispielaufgabe** – **Struktogramm**



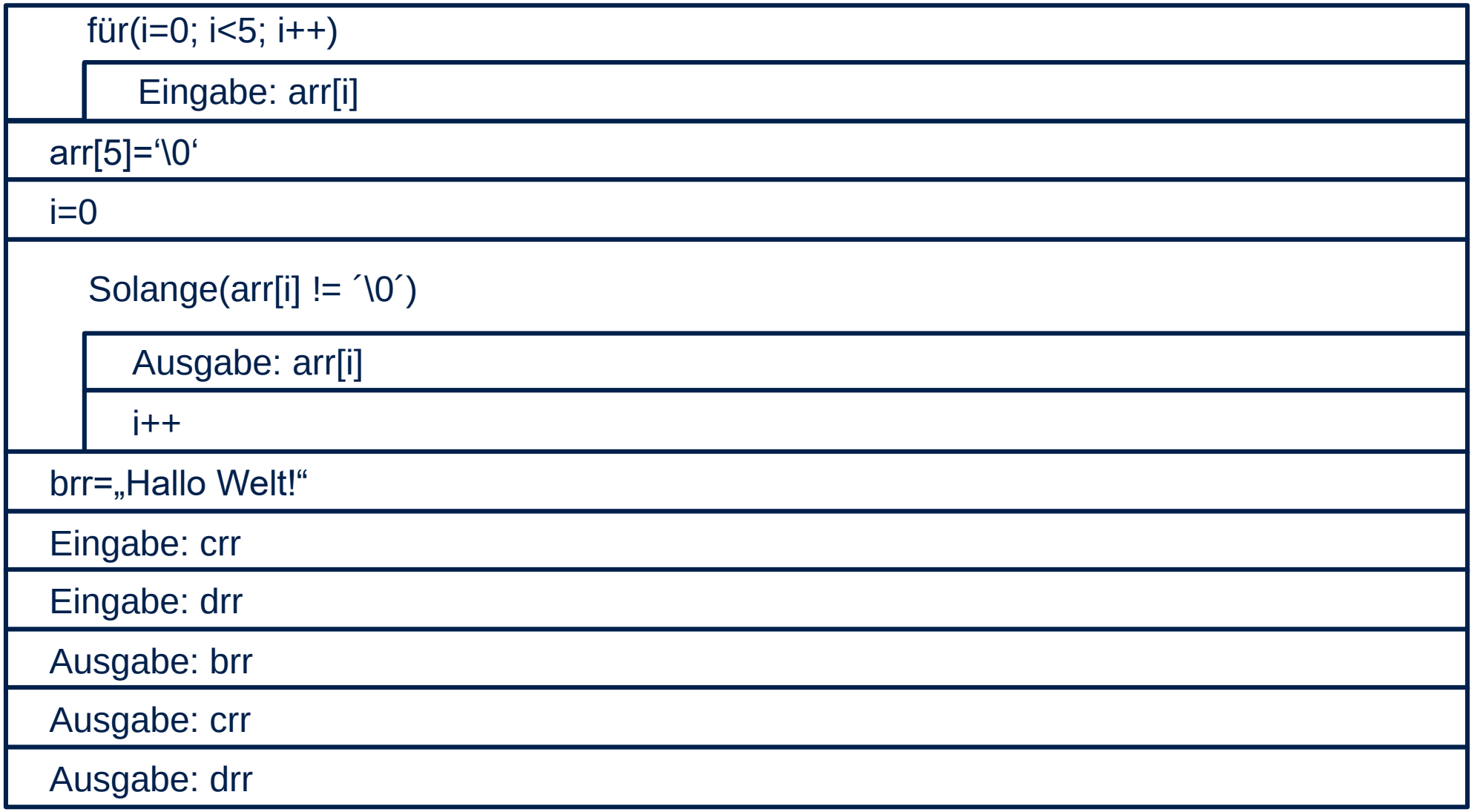
String (Zeichenkette) – **Beispielaufgabe** – **Struktogramm**



String (Zeichenkette) – **Beispielaufgabe** – **Struktogramm**



String (Zeichenkette) – **Beispielaufgabe** – **Struktogramm**



String (Zeichenkette) – **Beispielaufgabe** – **Pseudocode**

```
Programm „String-Beispiel“
{
        für(i=0;i<5;i++)
        {
                Eingabe: arr[i]
        }
        arr[5]=‘\0‘

        i=0
        while(arr[i] != ‘\0‘)
        {
                Ausgabe: arr[i]
                i++
        }

        brr=„Hallo Welt!“
        Eingabe: crr
        Eingabe: drr

        Ausgabe: brr
        Ausgabe: crr
        Ausgabe: drr
}
```

String (Zeichenkette) – **Beispielaufgabe** – Quellcode

```
#include <stdio.h>
#include <windows.h>
#include <string.h>

main()
{
   int i;
   char arr[10],brr[20],crr[15],drr[50];

   for(i=0;i<5;i++)
   {
      printf("Geben Sie bitte den %d. Buchstaben ein: ",i+1);
      fflush(stdin);
      scanf("%c",&arr[i]);
   }
   arr[5]='\0';

   i=0;
   while(arr[i]!='\0')
   {
      printf("%c",arr[i]);
      i++;
   }

   // Die Funktion strcpy (STRingCoPY) ersetzt den Zuweisungsoperator, denn ...
   // ... brr="Hallo Welt!" ist nicht erlaubt, da ein String in ANSI C kein Typ ist
   strcpy(brr,"Hallo Welt!");

   // Eingabe eines Strings OHNE Leerzeichen
   printf("\n\nGeben Sie bitte ein Wort ein: ");
   fflush(stdin);
   scanf("%s",crr); // crr ist die Abkürzung für &crr[0] (Adresse des Feldes crr[0])

   // Eingabe eines Strings MIT Leerzeichen
   printf("\nGeben Sie bitte einen Satz ein: ");
   fflush(stdin);
   scanf("%[^\n]",drr); // [^\n] liest alle Zeichen ein (auch Leerzeichen) mit Ausnahme von '\n' (Umbruch)

   // Ausgabe mit printf:
   printf("\nString in brr=%s\nString in crr=%s\nString in drr=%s\n\n\n",brr,crr,drr);

   system("pause");
}
```

WBS TRAINING

String (Zeichenkette) – **Beispielaufgabe** – Terminator

```
#include <stdio.h>
#include <windows.h>
#include <string.h>

main()
{
   int i;
   char arr[10],brr[20],crr[15],drr[50];

   for(i=0;i<5;i++)
   {
      printf("Geben Sie bitte den %d. Buchstaben ein: ",i+1);
      fflush(stdin);
      scanf("%c",&arr[i]);
   }
      arr[5]='\0';

   i=0;
   while(arr[i]!='\0')
   {
      printf("%c",arr[i]);
      i++;
   }


    // Die Funktion strcpy (STRingCoPY) ersetzt den Zuweisungsoperator, denn ...
   // ... brr="Hallo Welt!" ist nicht erlaubt, da ein String in ANSI C kein Typ ist
   strcpy(brr,"Hallo Welt!");

   // Eingabe eines Strings OHNE Leerzeichen
   printf("\n\nGeben Sie bitte ein Wort ein: ");
   fflush(stdin);
   scanf("%s",crr); // crr ist die Abkürzung für &crr[0] (Adresse des Feldes crr[0])

   // Eingabe eines Strings MIT Leerzeichen
   printf("\nGeben Sie bitte einen Satz ein: ");
   fflush(stdin);
   scanf("%[^\n]",drr); // [^\n] liest alle Zeichen ein (auch Leerzeichen) mit Ausnahme von '\n' (Umbruch)

   // Ausgabe mit printf:
   printf("\nString in brr=%s\nString in crr=%s\nString in drr=%s\n\n\n",brr,crr,drr);

   system("pause");
}
```

WBS TRAINING

String (Zeichenkette) – **Beispielaufgabe** – **string.h**

```
#include <stdio.h>
#include <windows.h>
#include <string.h>

main()
{
   int i;
   char arr[10],brr[20],crr[15],drr[50];

   for(i=0;i<5;i++)
   {
      printf("Geben Sie bitte den %d. Buchstaben ein: ",i+1);
      fflush(stdin);
      scanf("%c",&arr[i]);
   }
   arr[5]='\0';

   i=0;
   while(arr[i]!='\0')
   {
      printf("%c",arr[i]);
      i++;
   }

   // Die Funktion strcpy (STRingCoPY) ersetzt den Zuweisungsoperator, denn ...
   // ... brr="Hallo Welt!" ist nicht erlaubt, da ein String in ANSI C kein Typ ist
   strcpy(brr,"Hallo Welt!");

   // Eingabe eines Strings OHNE Leerzeichen
   printf("\n\nGeben Sie bitte ein Wort ein: ");
   fflush(stdin);
   scanf("%s",crr); // crr ist die Abkürzung für &crr[0] (Adresse des Feldes crr[0])

   // Eingabe eines Strings MIT Leerzeichen
   printf("\nGeben Sie bitte einen Satz ein: ");
   fflush(stdin);
   scanf("%[^\n]",drr); // [^\n] liest alle Zeichen ein (auch Leerzeichen) mit Ausnahme von '\n' (Umbruch)

   // Ausgabe mit printf:
   printf("\nString in brr=%s\nString in crr=%s\nString in drr=%s\n\n\n",brr,crr,drr);

   system("pause");
}
```

String (Zeichenkette) – **Beispielaufgabe** – strcpy

```
#include <stdio.h>
#include <windows.h>
#include <string.h>

main()
{
   int i;
   char arr[10],brr[20],crr[15],drr[50];

   for(i=0;i<5;i++)
   {
      printf("Geben Sie bitte den %d. Buchstaben ein: ",i+1);
      fflush(stdin);
      scanf("%c",&arr[i]);
   }
   arr[5]='\0';

   i=0;
   while(arr[i]!='\0')
   {
      printf("%c",arr[i]);
      i++;
   }

    // Die Funktion strcpy (STRingCoPY) ersetzt den Zuweisungsoperator, denn ...
    // ... brr="Hallo Welt!" ist nicht erlaubt, da ein String in ANSI C kein Typ ist
   strcpy(brr,"Hallo Welt!");

   // Eingabe eines Strings OHNE Leerzeichen
   printf("\n\nGeben Sie bitte ein Wort ein: ");
   fflush(stdin);
   scanf("%s",crr); // crr ist die Abkürzung für &crr[0] (Adresse des Feldes crr[0])

   // Eingabe eines Strings MIT Leerzeichen
   printf("\nGeben Sie bitte einen Satz ein: ");
   fflush(stdin);
   scanf("%[^\n]",drr); // [^\n] liest alle Zeichen ein (auch Leerzeichen) mit Ausnahme von '\n' (Umbruch)

   // Ausgabe mit printf:
   printf("\nString in brr=%s\nString in crr=%s\nString in drr=%s\n\n\n",brr,crr,drr);

   system("pause");
}
```

WBS TRAINING

String (Zeichenkette) – **Beispielaufgabe** – scanf OHNE Leerzeichen

```
#include <stdio.h>
#include <windows.h>
#include <string.h>

main()
{
   int i;
   char arr[10],brr[20],crr[15],drr[50];

   for(i=0;i<5;i++)
   {
      printf("Geben Sie bitte den %d. Buchstaben ein: ",i+1);
      fflush(stdin);
      scanf("%c",&arr[i]);
   }
   arr[5]='\0';

   i=0;
   while(arr[i]!='\0')
   {
      printf("%c",arr[i]);
      i++;
   }

   // Die Funktion strcpy (STRingCoPY) ersetzt den Zuweisungsoperator, denn ...
   // ... brr="Hallo Welt!" ist nicht erlaubt, da ein String in ANSI C kein Typ ist
   strcpy(brr,"Hallo Welt!");

   // Eingabe eines Strings OHNE Leerzeichen
   printf("\n\nGeben Sie bitte ein Wort ein: ");
   fflush(stdin);
   scanf("%s",crr); // crr ist die Abkürzung für &crr[0] (Adresse des Feldes crr[0])

   // Eingabe eines Strings MIT Leerzeichen
   printf("\nGeben Sie bitte einen Satz ein: ");
   fflush(stdin);
   scanf("%[^\n]",drr); // [^\n] liest alle Zeichen ein (auch Leerzeichen) mit Ausnahme von '\n' (Umbruch)

   // Ausgabe mit printf:
   printf("\nString in brr=%s\nString in crr=%s\nString in drr=%s\n\n\n",brr,crr,drr);

   system("pause");
}
```

WBS TRAINING

String (Zeichenkette) – **Beispielaufgabe** – scanf MIT Leerzeichen

```
#include <stdio.h>
#include <windows.h>
#include <string.h>

main()
{
   int i;
   char arr[10],brr[20],crr[15],drr[50];

   for(i=0;i<5;i++)
   {
      printf("Geben Sie bitte den %d. Buchstaben ein: ",i+1);
      fflush(stdin);
      scanf("%c",&arr[i]);
   }
   arr[5]='\0';

   i=0;
   while(arr[i]!='\0')
   {
      printf("%c",arr[i]);
      i++;
   }

   // Die Funktion strcpy (STRingCoPY) ersetzt den Zuweisungsoperator, denn ...
   // ... brr="Hallo Welt!" ist nicht erlaubt, da ein String in ANSI C kein Typ ist
   strcpy(brr,"Hallo Welt!");

   // Eingabe eines Strings OHNE Leerzeichen
   printf("\n\nGeben Sie bitte ein Wort ein: ");
   fflush(stdin);
   scanf("%s",crr); // crr ist die Abkürzung für &crr[0] (Adresse des Feldes crr[0])

   // Eingabe eines Strings MIT Leerzeichen
   printf("\nGeben Sie bitte einen Satz ein: ");
   fflush(stdin);
   scanf("%[^\n]",drr); // [^\n] liest alle Zeichen ein (auch Leerzeichen) mit Ausnahme von '\n' (Umbruch)

   // Ausgabe mit printf:
   printf("\nString in brr=%s\nString in crr=%s\nString in drr=%s\n\n\n",brr,crr,drr);

   system("pause");
}
```

WBS TRAINING

String (Zeichenkette) – **Beispielaufgabe** – printf(“%s“)

```
#include <stdio.h>
#include <windows.h>
#include <string.h>

main()
{
   int i;
   char arr[10],brr[20],crr[15],drr[50];

   for(i=0;i<5;i++)
   {
      printf("Geben Sie bitte den %d. Buchstaben ein: ",i+1);
      fflush(stdin);
      scanf("%c",&arr[i]);
   }
   arr[5]='\0';

   i=0;
   while(arr[i]!='\0')
   {
      printf("%c",arr[i]);
      i++;
   }

   // Die Funktion strcpy (STRingCoPY) ersetzt den Zuweisungsoperator, denn ...
   // ... brr="Hallo Welt!" ist nicht erlaubt, da ein String in ANSI C kein Typ ist
   strcpy(brr,"Hallo Welt!");

   // Eingabe eines Strings OHNE Leerzeichen
   printf("\n\nGeben Sie bitte ein Wort ein: ");
   fflush(stdin);
   scanf("%s",crr); // crr ist die Abkürzung für &crr[0] (Adresse des Feldes crr[0])

   // Eingabe eines Strings MIT Leerzeichen
   printf("\nGeben Sie bitte einen Satz ein: ");
   fflush(stdin);
   scanf("%[^\n]",drr); // [^\n] liest alle Zeichen ein (auch Leerzeichen) mit Ausnahme von '\n' (Umbruch)

   // Ausgabe mit printf:
   printf("\nString in brr=%s\nString in crr=%s\nString in drr=%s\n\n\n",brr,crr,drr);

   system("pause");
}
```

WBS TRAINING

Stringfunktionen – Definition und Motivation

- Wir haben mit **strcpy** bereits eine erste Stringfunktion kennengelernt.
- Ebenfalls haben wir bereits erfahren, dass sich Stringfunktionen in der Bibliothek **string.h** befinden.
- Auch wurde bereits eine erste Motivation für Stringfunktionen angesprochen:
  - Zum einen werden Funktionalitäten angeboten, die uns bei „normalen Variablen“ zur Verfügung stehen:
    - **strcpy** ersetzt – wie bereits gezeigt – den **Zuweisungsoperator**
    - **strcmp** ersetzt (wie wir uns im Anschluss näher anschauen werden) die Operatoren **‘==‘**, **‘<‘** und **‘>‘**
  - Zum anderen aber gibt es auch Funktionalitäten, die nur für Strings von Bedeutung sind:
    - **strlen** ermittelt die **Länge** eines Strings
    - **strcat** erlaubt das „**Aneinanderhängen**“ bzw. „Zusammenfügen“ von Strings

Zum Abschluss dieser Vorlesung nun also noch eine nähere Betrachtung der Stringfunktionen **strcmp, strlen** und **strcat**

Stringfunktionen – strcmp

- strcmp(“Aachen“, “Zürich“) = **-1**

  (alphabetisch korrekt => Rückgabewert: -1)

- strcmp(“Zürich“, “Aachen“) = **1**

  (umgekehrt alphabetisch => Rückgabewert: 1)

- strcmp(“Aachen“, “Aachen“) = **0**

  (identische Strings => Rückgabewert: 0)

Stringfunktionen – strlen

- strlen(“abc“) = **3**

  (Es werden die Zeichen gezählt – OHNE Terminator)

- strlen(““) = **0**

  (Ein „Leerstring“, der also nur aus einem Terminator besteht, hat entsprechend die Länge 0)

- strlen(“Hallo Welt!“) = **11**

  (Leerzeichen werden mitgezählt)

Stringfunktionen – strcat

Zunächst einige „Vorarbeiten“:

strcpy(string1,“Hallo “);
strcpy(string2,“Welt!“);

Kontrollausgabe:

printf(“string1=%s string2=%s“,string1,string2);

```
string1=Hallo  string2=Welt!
```

Anwendung von strcat:

strcat(string1, string2);

Kontrollausgabe:

printf(“string1=%s string2=%s“,string1,string2);

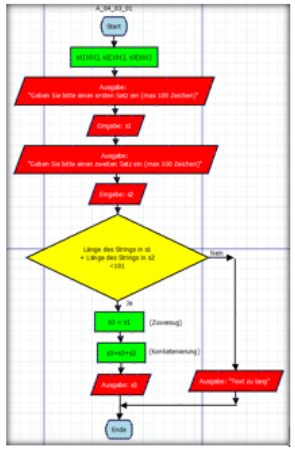
```
string1=Hallo Welt! string2=Welt!
```

String (Zeichenkette) – Gemeinsame Übung A_04_03_01

WBS TRAINING

Aufgabe_04_03_01

Gegeben sei der folgende PAP:



Aufgabenstellung:
Bitte erstellen Sie dazu einen geeigneten **Quellcode** in ANSI C.

WBS TRAINING AG
Lorenzweg 5
D-12099 Berlin
Amtsgericht Berlin HRB 68531
Sitz der Gesellschaft: Berlin

Vorstand:
Heinrich Kronbichler,
Joachim Giese
Aufsichtsrat (Vorsitz): Dr. Daniel Stadler
USt-IdNr.: DE 209 768 248

GLS Gemeinschaftsbank eG
IBAN: DE18 4306 0967 1146 1814 00
BIC: GENODEM1GLS

WBS TRAINING

VIELEN DANK FÜR IHRE AUFMERKSAMKEIT!