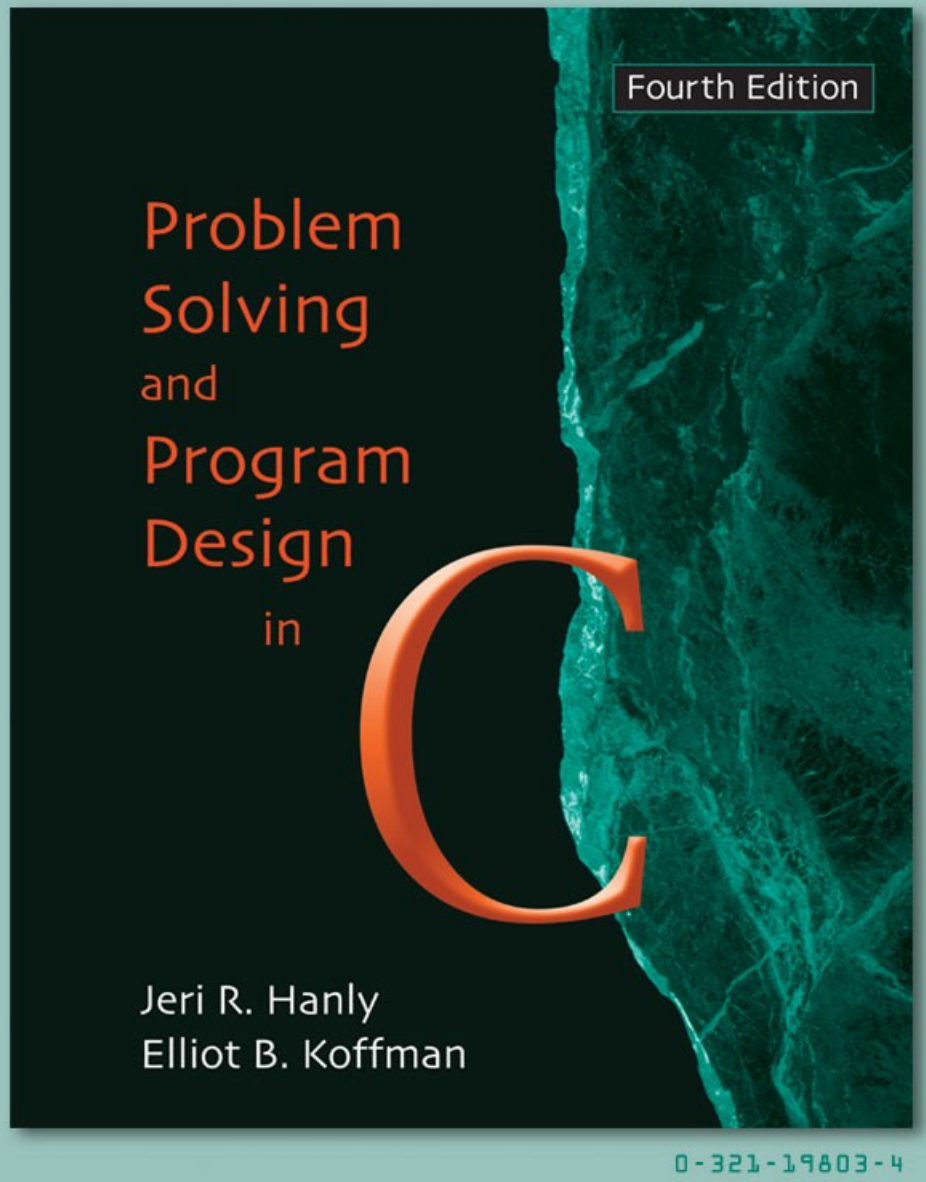
Strings



PEARSON
Addison
Wesley

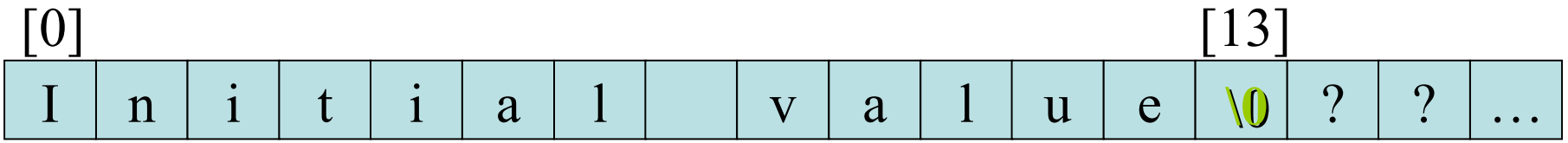
Strings

- C implements the **string** data structure using arrays of type `char`.
- You have already used the string extensively.
  - printf("This program is terminated!\n");

- Since **string** is an array, the declaration of a string :
  - char string [size];
    - char string_var[30];

Memory Storage for a String

- The string is always ended with a **null character '\0'**.
- The characters after the null character are ignored.
- e.g., char str[20] = "Initial value";



Strings initialization

- Char name[5]=”code”
- Char name[5]={‘c’,’o’,’d’,’e’,’\0’}
- Char name[]=”code”

/ Illegal

- **Char name[3]=”code”**
- **Char name[5]; name =”code”**

Arrays of Strings(Table)

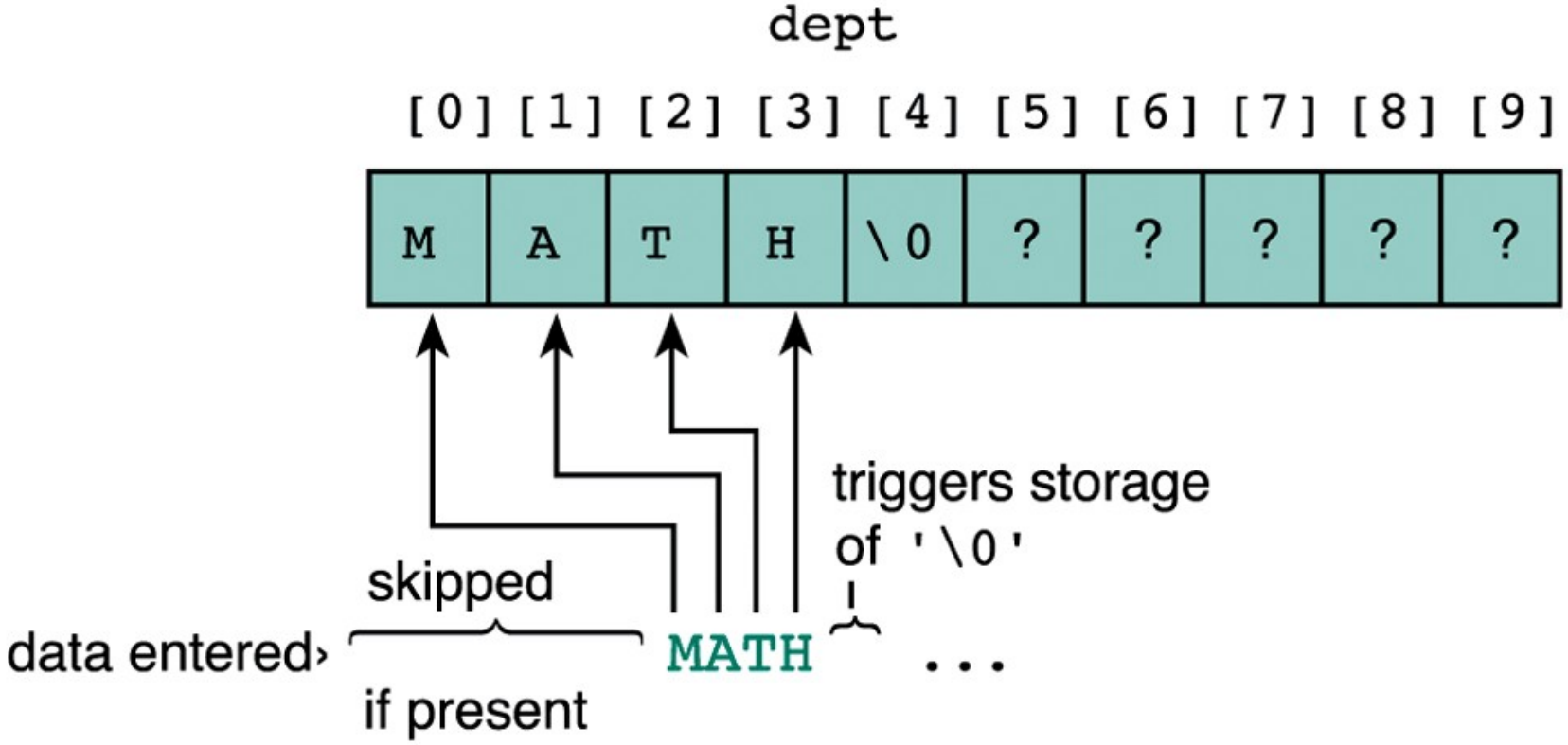
- `char month[5][10] = {“January”, “February”, “March”, “April”, “May”};`

```c
#include<stdio.h>
void  main ()
{
    char month[5][10];
    int i;
    for(i=0;i<5;i++)
      gets(month[i]);
}
```

J	a	n	u	a	r	y	\0		
F	e	b	r	u	a	r	y	\0	
M	a	r	c	h	\0				
A	p	r	i	l	\0				
M	a	y	\0						

Execution of scanf ("%s", dept);

- Whenever encountering a white space, the scanning stops and `scanf` places the null character at the end of the string.
- e.g, if the user types "MATH 1234 TR 1800," the string "MATH" along with '0' is stored into `dept`.



getchar() ,gets() ,putchar() and puts()

- getchar() terminates when ‘\n’ encounters

  Note: null character must insert at end

- gets() terminates when ‘\n’ encounters and append ‘\0’ at end automatically

String operations &Library Functions

- The string can not be copied by the assignment operator ‘=’.
  - “`str = “Test String”`” is not valid.
- C provides string manipulating functions in the “string.h” library.

Distinction Between Characters and Strings

- The representation of a char (e.g., ‘Q’) and a string (e.g., “Q”) is essentially different.

Character ‘Q’

String “Q”

Some String Functions from String.h

Function	Purpose	Example
`strcpy`	Makes a copy of a string	strcpy(s1, “Hi”);
`strcat`	Appends a string to the end of another string	strcat(s1, “more”);
`strcmp`	Compare two strings alphabetically	strcmp(s1, “Hu”);
`strlen`	Returns the number of characters in a string	strlen(“Hi”) returns 2.

String length

```c
#include<stdio.h>
void  main ()
{
      char str[20];
      int l;
      gets(str);
      for(l=0;str[l]!='\0';l++);
      printf("length=%d",l);

}
```

String copy

```c
#include<stdio.h>
void main()
{
    char str1[20],str2[20];
    int i;
    scanf("%s",str1);
    for (i=0;str1[i]!='\0';i++)
        str2[i]=str1[i];
    str2[i]='\0';
    puts(str2);
}
```

String concatenation

```
s3=s1+s2 ; //illegal
s3=s1+"hello" ;//illegal

#include<stdio.h>
void  main ()
{
    char str1 [20]="computer",str2 [20]="programming",str3[40];
    int i,j;
    for (i=0;str1 [i]!='\0';i++)
        str3[i]=str1[i];
    for (j=0;str2 [j]!='\0';j++)
        str3[i+j]=str2[j];

    str3 [i+j]='\0';
    printf("%s",str3);

}
```

String compare

```
if(name1==name2);  //not permitted
if(name=="ABC");  //not permitted
#include<stdio.h>
void  main ()
{
     char str1 [20],str2 [20];
     int i;
     gets(str1);
     gets(str2);
     while(str1[i]==str2[i] && str1[i]!='\0' && str2[i]!='\0')
          i++;
     if(str1[i]=='\0' && str2[i]=='\0')
          printf("strings are equal");
     else
          printf("strings are not euqal");

}
```

String Comparison

Relationship	Returned Value	Example
`str1 < str2`	Negative	“Hello”< “Hi”
`str1 = str2`	0	“Hi” = “Hi”
`str1 > str2`	Positive	“Hi” > “Hello”

ASCII value of first unmatched character

- e.g., we can check if two strings are the same by

```
if(strcmp(str1, str2) != 0)
  printf(“The two strings are different!”);
```

Character Analysis and Conversion

- The **<ctype.h>** library defines facilities for character analysis and conversion.

Functions	Description
`isalpha`	Check if the argument is a letter
`isdigit`	Check if the argument is one of the ten digits
`isspace`	Check if argument is a space, newline or tab.
`tolower`	Converts the lowercase letters in the argument to upper case letters.

```c
#include <stdio.h>
#include <ctype.h>
int main()
{
    char c;
    c = 'Q';
    printf("\nResult when uppercase alphabet is passed: %d", isalpha(c));

    c='+';
    printf("\nResult when non-alphabetic character is passed: %d", isalpha(c));

    return 0;
}
```

```
Result when uppercase alphabet is passed: 1024
Result when non-alphabetic character is passed: 0
Process returned 0 (0x0)   execution time : 0.001 s
Press ENTER to continue.
```