Reading and Writing String

Reading Strings

- Format conversion "/s" can be used in scanf for reading strings not containing white spaces: scanf("%s", str)
- ' & ' not required before str as it is a pointer.
- C string library provides gets(str) to read a string.
- It does not skip white spaces like scanf does.
- But, scanf and gets have no way to know the size of the array in advance.
- So potentially, they could attempt to store character past the array.
- scanf can use "%.ns" conversion to avoid this problem.
- But gets is dangerous.
Reading and Writing String

Reading Character by Character

- Since both `scanf` and `gets` are potentially risky, character by character reading using `getchar`, and similarly writing using `putchar` are safer to use.

- Code for `readLine` for example will be:

```c
int readLine(char str[], int n) {
    int ch, i = 0;

    while ((ch = getchar()) != '\n') {
        if (i < n)
            str[i++] = ch;
    }
    str[i] = '\0';
    return i;
}
```
Reading and Writing String

Accessing Characters in a String

```c
int countWhiteSpace(const s[]) {
    int cnt = 0, i;
    for (i = 0; s[i] != ' \0'; i++) {
        if (s[i] == ' ')
            cnt++;
    }
    return cnt;
}
```

```c
int countWhiteSpaces(const char *s) {
    for (; *s != ' \0', s++)
        if (*s == ' ')
            cnt++;
    return cnt;
}
```
Manipulating Strings

String Functions

- Direct copy or compare strings result in failures.

```c
// Consider following code snippet
char strA[10], strB[10];

strA = "abc";
strB = strA; // ---WRONG--- (assignment of incompatible types)
if (strA == strB) // ---WRONG--- (pointers compared.) returns 0
```
Manipulating Strings

String Functions

- Then how do we do it? C provide rich set of string functions.

- `#include <string.h>` to access these string manipulation functions.

- `strlen(s)`: returns length of the string.

- `strcpy(dest,src)`: copies one string into another.

- `strcat(dest,src)`: appends one string to another.

- `strcmp(first,second)`: return 1 if equal 0 otherwise.
Manipulating Strings

Length of a String

```c
#include <stdio.h>
int readLine(char str[], int n);
int len(const char *s);

int main() {
    char str[50];
    readLine(str, 49);
    printf("Length = %d\n", len(str));
}

int len(const char *s) {
    int n = 0;
    for (; *s; s++) // when *s = '\0' loop exits
        n++;
    return n;
}
```
Examples for String Manipulations

Example

Example (1)

- Suppose a string stores a name in the form "First/Last".
- We need a function to find "/" in the name.
- There is a C function `strchr` that can be used.
- But we want to write a function which takes a pointer to string and a character and returns a pointer to the separator.
Examples for String Manipulations

Example

Example (1 contd)

```c
#include <stdio.h>
#include <string.h>
char * firstLast(char *s, char ch) {
    char *p = s;
    while (*p != ch) {
        if (*p == '\0')
            return NULL;
        p++;
    }
    return p;
}
```
Examples for String Manipulations

Example

Example (1 contd.)

```c
int main() {
    char str[] = "Deigo/Maradona";
    char *ptr, *q, c = '/';
    int l = strlen(str);
    ptr = firstLast(str, c);
    for (q = str; q < ptr; q++)
        printf("%c", *q);
    printf("\n");
    for (q = ptr+1; q < str + l; q++)
        printf("%c", *q);
    printf("\n");
}
```
Examples for String Manipulations

Example

Example (2)

Let us write the following program:

- After user enter words, program determines which words comes first and last if the words were listed in dictionary order.
- The programs stop accepting words once the user enters a four-letter word.

1. Initially read one word, define it both as the smallest and the largest.
2. If the word read has 4 letters stop, and print the smallest and the largest.
3. Otherwise, read next word, update the smallest and the largest.
Examples for String Manipulations

Example (2 contd.)

```c
#include <stdio.h>
#include <string.h>
int main() {
    char smallest[20];    // stores minimum
    char largest[20];     // stores maximum
    char word[20];        // stores current word
    char ch;
    int i = 0;

    printf("Enter the words\n");
    /*** Rest of the code ***/
}
```
Examples for String Manipulations

Example (2 contd.)

```c
// Read the first word, set it as the smallest word
while ((ch = getchar()) != ' \n' && i < 19)
    smallest[i++] = ch;
smallest[i] = ' \0'; // end string

if (strlen(smallest) == 4) { // no more input
    printf("smallest word: %s \n", smallest);
    printf("largest word: %s \n", smallest);
    return; // stop
}

// Initially the smallest is also the largest
strcpy(largest, smallest);

/*** Rest of the code for reading more words ***/
```
Examples for String Manipulations

Example (2 contd.)

```c
while (1) {
    i = 0;
    while ((ch = getchar()) != '\n' && i < 19)
        word[i++] = ch;

    if (strcmp(word, smallest) < 0) // compare with smallest
        strcpy(smallest, word);      // current is smallest
    if (strcmp(word, largest) > 0) // compare with largest
        strcpy(largest, word);       // current is largest
    if (strlen(word) == 4)        // exit on a word with 4 letters
        break;
}
printf("smallest word:\%s\n", smallest); // print smallest
printf("largest word:\%s\n", largest);  // print largest
```
Examples for String Manipulations

Example (3)

Create a program for printing calendar reminders for events. Programs takes each reminder in form of a date (a digit number) and a string.

- Each reminder can be a string of at most 60 characters.
- Upto 50 reminders can be stored.
- End of input is signalled by 0.
Examples for String Manipulations

Example (3 input format)

```
Enter day and reminder: 26 Republic day dinner at 7.00PM
Enter day and reminder: 5 Movie - "A beautiful mind"
Enter day and reminder: 12 Dental appointment
Enter day and reminder: 5 Exta class
Enter day and reminder: 16 Make up lab
Enter day and reminder: 0
```
Examples for String Manipulations

Example (3 output Format.)

The program should output the following:

Day Reminder
  5 Extra class
  5 Movie - "A beautiful mind"
12 Dental appointment
16 Make up lab
26 republic day dinner at 7.00PM
Examples for String Manipulations

Example (3 code)

- The code is fragmented into 5 parts.
- First part is for reading a line of text and storing it as a string.
- Second part specifies declaration of all needed variables
- Third and fourth part implement main logic of reading a line storing it in sorted order.
- Final part is for printing out the stored reminders.
Examples for String Manipulations

The code appears in slide No. 12 also.

Example (3: reading a line of text)

```c
int readline(char str[], int n) {
    int ch, i = 0;

    while (((ch = getchar()) != '
')) {
        if (i < n)
            str[i++] = ch;
    }
    str[i] = '\0';
    return i;
}
```
Examples for String Manipulations

Example (3: variables)

```c
#include <stdio.h>
#include <string.h>
define MAX_REMIND 50
define MSG_LEN 60

int readLine(char str[], int n);

int main() {
    char reminders[MAX_REMIND][MSG_LEN + 3];
    char day_str[3], msg_str[MSG_LEN + 1];
    int day, i, j, n_reminders = 0;

    /*** Rest of the program ***/
}
```
Examples for String Manipulations

Example (3 code: main loop)

```c
for (;;) {
    if (n_reminders == MAX_REMIND) {
        printf("No space for insertion \n");
        break;
    }
    printf("Enter day and reminder: ");
    scanf("%2d", &day); // read the day
    if (day == 0) // 0 input --> no more reminders
        break;

    // write day into character array day_str
    sprintf(day_str, "%2d", day);

    //*** Other part of for loop on next slide ***/
}
```