CS10003:
Programming & Data Structures

Dept. of Computer Science & Engineering
Indian Institute of Technology Kharagpur

Autumn 2020

Character Arrays and Strings

char C[8] = { 's', 'w', 'a', 'g', 'a', 't', 'o', '\0' };

- C[0] gets the value 'a', C[1] the value 'b', and so on. The last (7th) location receives the null character ‘\0’
- Null-terminated (last character is ‘\0’) character arrays are also called strings
- Strings can be initialized in an alternative way. The last declaration is equivalent to:

  char C[8] = "swagato";

- The trailing null character is missing here. C automatically puts it at the end if you define it like this
- Note also that for individual characters, C uses single quotes, whereas for strings, it uses double quotes

Reading strings: %s format

```
int main()
{
    char name[25];
    scanf("%s", name);
    printf("Name = %s \n", name);
    return 0;
}
```

%s reads a string into a character array given the array name or start address.

It ends the string with ‘\0’

An example

```
int main()
{
  #define SIZE 25
  int i, count=0;
  char name[SIZE];
  scanf("%s", name);
  printf("Name = %s \n", name);
  for (i=0; name[i]!='\0'; i++)
    if (name[i] == 'a') count++;
  printf("Total a's = %d\n", count);
  return 0;
}
```

Seen on screen

Typed as input

Satyanarayana

Name = Satyanarayana

Total a's = 6

Printed by program

Note that character strings read in %s format end with '\0'

Palindrome Checking

```
int main()
{
   const int SIZE = 25;
   int i, flag, count=0;
   char name[SIZE];
   scanf("%s", name);     /* Read Name */
   for (i=0; name[i]!='\0'; i++);    /* Find Length of String */
   printf("Total length = %d\n",i);
   count=i;  flag = 0;
   /* Loop below checks for palindrome  by comparison*/
   for(i=0; i<count; i++) if (name[i]!=name[count-i-1]) flag = 1;
   if (flag ==0) printf ("%s is a Palindrome\n", name);
   else printf("%s is NOT a Palindrome\n", name);
   return 0;
}
```

Some exercises

1. Write a C program that reads an integer n and stores the first n Fibonacci numbers in an array.
2. Write a C program that reads an integer n and uses an array to efficiently find out the first n prime numbers.
3. Read in an integer n, read in n integers and print the integer with the highest frequency.
4. Read in an integer n, read in n numbers and find out the mean, median and mode.
5. Read in two names and compare them and print them in lexicographic (dictionary) order.
6. Read in an integer n, read in n names and print the last name when compared in lexicographic order.