Tutorial Sheet (September 07, 2018)

ESC101 – Fundamentals of Computing

Announcements

1. **Make-up lab for B4, B5, B6, B13,** Sep 14 (today) CC-01,02, 2PM
2. **Mid-sem theory exam**, September 22nd, 2018 (Saturday) 1PM – 3PM (afternoon). Assigned seating – will be made known to you.
3. Theory exam is open handwritten notes. Printouts, photocopies, use of mobile phones, iPads and other communication devices prohibited – will be considered cheating.
4. Bring a pencil, eraser, sharpener with you – just like major quiz.
5. Bring your IITK ID Card – keep your mobile phone, iPad at home
6. **Doubt clearing session**: Sep 15 (Saturday) CC-02 5PM – come prepared with your doubts. Students not comfortable with English welcome.
7. **Advanced track**: meet mentors and finalize topic asap.

Revision (ask for doubts)

1. **Arrays:** 0-based indexing, illegal array subscripts, segmentation faults, using (typecasted) integer expressions to address arrays, array initialization along with declaration.
2. **Characters**: stored internally as integers between 0 and 127 (ASCII value) e.g. 'A' is stored as 65, 'B' as 66, etc … 'a' as 97, 'b' as 98 etc …'0' as 48, '1' as 49 etc …, scanf, printf using %c, getchar, putchar. Care while mixing %c, %d in the same format string.
3. **Character arrays and strings**: char array with \0 (null) character is called a string. The word string is not a datatype or even a keyword in C. Strings can be initialized in bulk during declaration (not afterward) as char c[50] = "Good Morning!". scanf, printf using %s (**no & needed in scanf**), gets, puts.

4. **Delimited strings**: \0 (ASCII value 0) acts as a delimiter for strings. Anything stored beyond the first \0 is neglected by Mr C while printing and processing character arrays as strings. \0 automatically inserted by Mr C when we do bulk initialization, gets or scanf.
5. **Segfaults with strings**: If character array not long enough (number of user characters + 1 for delimited \0), then gets, scanf can cause segfault. If \0 missing from char array then even printf can cause segfault (by trying to read beyong end of array).

Sample Questions to discuss

Print all the letters of the alphabet with alternating letters in upper and lower case.

```
char c;
for(c = 'a'; c <= 'z'; c++)
   if((c – 'a') %2 == 0)
      printf("%c", c);
   else
      printf("%c", c – 'a' + 'A');
```

Write a code to safely read all characters from the input till the user enters the new line character, or else till the array is finished, whichever is earlier.

This is basically a safe version of gets and scanf which will not cause segfault if user enters too many characters.

```
char c, str[50];
int i
for(i = 0; i < 49; i++)
   // Reserve location for \0
   if((c = getchar()) != '\n')
      str[i] = c;
   else
      break;
str[i] = '\0';
```

Write code to take a string, reverse it and print it. Be careful to take care of the \0 character.

Discuss the case of lastLocation = -1, 0, 1, 2, 3, 4 to show that the integer division and the limits i=0 till i <= lastLocation/2 are indeed appropriate.

```
char c, str[50];
int i, lastLocation;
gets(str);
for(i = 0; i < 50; i++)
   if(str[i] == '\0')
      break;
lastLocation = i-1;
for(i = 0; i <= lastLocation/2; i++){
   c = str[i];
   str[i] = str[lastLocation – i];
   str[lastLocation – i] = c;
}
puts(str);
```

Some Pitfalls and recognizing compiler error messages

1. Array subscripts cannot be negative, nor can they be equal or greater to the length of the array.
2. Bulk initialization of arrays possible only during declaration, not afterward. Afterward, element-by-element initialization possible.
3. Arrays cannot be assigned, compared in bulk – need to write a loop to compare them element-by-element.
4. Don't bulk-initialize array with more elements during declaration with more elements than length of array – segmentation fault.
5. Prutor will give runtime errors on segfaults
6. '5' is a character representing the digit 5 (stored internally as the integer 53) and 5 is an integer with the value 5 itself.
7. char a = "d" will cause unexpected behaviors, compiler warning Should use char a = 'd' to initialize with character constant or else use typecasting char a = (char)('a' + 3);
