Formatted Input and Output

November 17, 2011

Outline

Streams

- `stdout` - The output stream
- `stdin` - The input stream
- `stderr` - The error stream

Streams

- `printf(‘‘Text’’);` is the same as `fprintf(stdout,‘‘Text’’);`

What is formatted input and output?

- Formatting scanned and printed values of variables
- What you already know:
  1. `printf`
  2. `scanf`

Examples

- Examples using `printf`, `scanf`
- `printf(“%d %f %lf”, 4525, 236.424, 246463.43226);`

Printing ints

- We can format integers to be printed as (un)signed decimals, octals and hexadecimals.
- Conversion specifiers are:
  1. d and i
  2. u
  3. x and X
  4. o
  5. h and l

Printing floats and doubles

- f
- e and E
- g and G
- %f specifies a float
- %lf specifies a double
- %Lf specifies a long double

Some more examples

- `printf(“%e %E %g %g”, 265.734, 265.734, 2.42, 26246464.3264);` outputs `2.657340e+02 2.657340E+02 2.42 2.62465e+07`
- `printf(“%p”, t);` outputs the value of `t`, where `t` is a pointer. Such a value usually looks like `0xbfa02ffc`

Some more specifiers

- `-`
- `+`
- *space*
- `#`

Escape characters

- What are escape characters?
- Characters in a sequence of characters that are used to provide an alternate interpretation.

An example

```
printf(‘‘                    This line has missing
characters \r’’);
printf(‘‘This adds some.  \n’’);
```

outputs

```
This adds some.  This line has missing characters
```

Using conversion specifiers with scanf

Conversion specifiers work pretty much the same way with `scanf` as they do with `printf`, a notable difference being the `i` specifier - it is used to read a decimal, octal, or hexadecimal integer.

Scan Sets

- Scan a stream of characters only till some specified characters are encountered.
- `scanf("%[abc]", s);` scans and stores characters into the string `s` until a character other than `a`, `b` or `c` occurs
- Use `%[^abc]` instead, to scan till either `a`, `b` or `c` are encountered.
- So, `scanf("%[^\n]", s);` scans characters till an end-of-line is encountered, and stores them in `s`.

Suppression character (*)

- Used to ignore data.
- Data read using this is immediately discarded, without storing it into a variable.
- `scanf(“%d %*d %d”, &a, &b);` scans three integers, but stores only the first and the third in variables `a` and `b`, respectively. The second number is scanned and ignored, i.e., not stored in any variable.