Software Tools

C, Unix (Linux), and tools

Variable Length Argument List

- Standard library printf (and its sisters) has a variable length argument list
- C provides a formal mechanism for writing such functions
- They are not used very often, but are very helpful

How it’s done

- File stdarg.h contains a set of macros for this purpose, plus a data type called va_list
- They are portable, although they me be implemented in a very different way in every system

```
int fun(int arg1, ...);
va_list ap;
va_start(ap, arg1);
va_arg(ap, int); or va_arg(ap, double);
va_end(ap);
```

Example: minprintf

- A very simplified printf from the book

```
void minprintf(char *fmt, ...)
{
  va_list ap;
  /* .... */
  va_start(ap, fmt);
  for (...){
    ...
    Ival = va_arg(ap, int);
    Dval = va_arg(ap, double);
  }
  va_end(ap);
}
```

Files in the standard library

- Files are handled through the standard I/O library
- The library provides a portable uniform and convenient way to handle files

```
#include <stdio.h>

FILE *fopen(const char *pathname, const char *mode);
int fclose(FILE *stream);
size_t fread(void *ptr, size_t size, size_t nmemb, FILE *stream);
size_t fwrite(const void *ptr, size_t size, size_t nmemb,
                      FILE *stream);
int ferror(FILE *stream);
```

Standard open files

- Every C program starts life with three files:
  - stdin
  - stdout
  - stderr

Suicides of programs

- If a process wants to terminate it calls exit.
  - A happy process calls exit(0)
  - An unhappy program calls exit(1) or exit(2)
  - In the main program it can also just issue return 0 or return 1, etc.
- A call to exit deallocates everything and releases any resources. Also kills its children!
  - Linux is not meant to provide good family values.

Height of a Tree

- As always in a .c file

```
int heightbintree(bintree t)
{
  int hl, hr;
  if (t==NULL) return 0;
  hl = heightbintree(t->l);
  hr = heightbintree(t->r);
  if (hl>hr)
    return hl+1;
  else
    return hr+1;
}
```

Standard I/O Library

- Contains many functions that help us with I/O
- All function declarations are in stdio.h which has to be included.
- These are functions that invoke system calls that do the actual I/O but are easier to use.
- These functions are meant to be portable, unlike the system calls that depend on the OS.

Getchar and Putchar

- Perform I/O character by character.

```
while ((c=getchar()) != EOF)
  putchar(isupper(c) ? tolower(c) : c);
```

Formated output

- Printf *et al*.
- Uses a formating string with formating directives
- Directives start with a % and end with a conversion character.
- After the % there might be a – (left alignment), a minimum field width, a . followed by the precision and a length modifier (l, letter ell)

Conversion Characters

```
d: decimal
o: octal
x: hexadecimal
u: unsigned
c: character
s: string
E, e: real number with exponential notation
f: decimal notation for reals (floats or doubles)
g: either %e or %f
```