

The C Info Sheet

C Expressions

Arithmetic operators:

+ - * / % (modulus)

Note: no exponentiation operator like ^ in Haskell

Relational operators:

== != < > <= >=

Logical operators:

! (NOT) && (AND) || (OR)

Bitwise operators:

~ (NOT) & (AND) | (OR) ^ (XOR)

<stdio.h> Library

FILE *f; - external file handle

int getchar(void),getc(FILE *f);
Fetch one character from input

int putchar(int ch),putc(int ch, FILE *f);
Display a character on output

char *gets(char *s);
char *fgets(char *s, int len, FILE *f);
Fetch one line into buffer

scanf(*format-string*, *address*₁, ...);
Interpret input according to formats;
store results in specified addresses;
return number of valid, stored values

e.g.

scanf("%d", &i); - read an integer
scanf("%f", &r); - read a real number
scanf("%c", &c); - read a character
scanf("%s", s); - read a string

printf(*format-string*, *expr*₁, ...);
Display expressions according to specified
formats; expression type must match format

e.g.

printf("%3d", i); - show an integer
printf("%.2f", r); - show a real number
printf("%c", c); - show a character
printf("%s", s); - show a string
printf("Hello\n"); - show literal text

<stdlib.h> Library

EOF - end of file marker for i/o

NULL - canonical invalid pointer

void *malloc(int size);
allocate memory and return address

void free(void *addr);
release previously allocated memory

int atoi(const char *nptr)
Converts string pointed to by nptr to int.

void exit(int status);
terminate program with return status

<ctype.h> Library

int isalpha(char), isdigit(char), isspace(char),
ispunct(char), islower(char), isupper(char);
character classification functions

char toupper(char), tolower(char);
case conversion functions

<math.h> Library

double sin(double), cos(double), tan(double);
trigonometric functions - inputs are radians

double log(double), pow(double, double);
double sqrt(double);
logarithms, raise to power, square root

<string.h> Library

char *strcpy(char *buff, char *src);
Copy src string into buffer buff

int strcmp(char *s1, char *s2);
Compare two strings; return difference

int strlen(char *s)
Return length of string s

char *strstr(char *s1, char *s2)
Check whether s2 occurs in s1;
return pointer to first occurrence in s1
of s2 or NULL if not present