**Cheat Sheet**

## Pointers:

Declare pointer to *type type* \*name

declare pointer to type *type* \*name  
declare function returning pointer to *type type* \*f()  
declare pointer to function returning *type type* (\*pf)()  
generic pointer type void \*  
null pointer NULL  
object pointed to by *pointer*  \**pointer*

address of object *name* &*name*

## Structs:

Typedef struct *tag*{*declarations*}*name*;

create structure struct *tag name*  
member of structure from template *name.member*  
member of pointed to structure *pointer* -> *member*  
Example. (\*p).x and p->x are the same

## Operators:

Size of an object *sizeof*

Conditional expression *test\_exp* ? *exp1* : *exp2*

## Character Class Tests <ctype.h>:

alphanumeric? isalnum(c)  
alphabetic? isalpha(c)  
control character? iscntrl(c)  
decimal digit? isdigit(c)  
printing character (not incl space)? isgraph(c)  
lower case letter? islower(c)  
printing character (incl space)? isprint(c)  
printing char except space, letter, digit? ispunct(c)  
space, formfeed, newline, cr, tab, vtab? isspace(c)  
upper case letter? isupper(c)  
hexadecimal digit? isxdigit(c)  
convert to lower case tolower(c)  
convert to upper case toupper(c)

## String Operations <string.h>:

*s,t are strings, cs,ct are constant strings*  
length of s strlen(s)  
copy ct to s strcpy(s, ct) (destination, source)  
up to n chars strncpy(s, ct, n)  
concatenate ct after s strcat(s, ct) (add ct to end of s)  
up to n chars in ct strncat(s, ct, n)

compare cs to ct strcmp(cs, ct)

|  |  |  |
| --- | --- | --- |
| str1 less than str2 | Negative number | strcmp(str1, str2) < 0 |
| str1 equal to str2 | 0 | strcmp(str1, str2) == 0 |
| str1 greater than str2 | Positive number | strcmp(str1, str2) > 0 |

only first n chars strncmp(cs, ct, n)  
pointer to first c in cs strchr(sourceStr, searchChar)

Returns NULL if searchChar does not exist in sourceStr. Else, returns pointer to first occurrence.  
pointer to last c in cs strrchr(sourceStr, searchChar)

Returns NULL if searchChar does not exist in sourceStr. Else, returns pointer to LAST occurrence (searches in reverse, hence middle 'r' in name).  
char pointer to substring strstr(str1, str2)

Returns NULL if str2 does not exist in str1. Else, returns a char pointer pointing to the first character of the first occurrence of string str2 within string str1.

## Malloc, Realloc, Free:

pointerVariableName = (type\*)malloc(sizeof(type));

free(pointerVariable);

pointerVariable = (type\*)realloc(pointerVariable, numElements \* sizeof(type))

## Strtok:

Tokenize string char \*name = strtok(string, delimiter);

If wishing to use string from previously ran strtok, use NULL instead of string

## Files:

Stdin, stdout

Fprintf(FILE\*, “Format string”, vars to be printed);

Fscanf(FILE\*, “Format string”, location for val to be stored);

Sscanf(string, “format string”, location for val to be stored); used to read a sequence of characters from a c string

fgets(char \*str, int n, FILE \*stream); Reads a whole line/string

FILE\* inFile = NULL; // File pointer

inFile = fopen("myfile.txt", "r"); (w if you want to write to file) returns NULL if file DNE

fclose(inFile); close file at the end of program

feof() returns 1 if the previous read operation reached the end of the file

## Output Formatting:

%(flags)(width)(.precision)specifier

Width: Specifies the minimum number of characters to print. If the formatted value has more characters than the width, the value will not be truncated. If the formatted value has fewer characters than the width, the output will be padded with spaces (or 0's if the '0' flag is specified).

.precision Specifies the number of digits to print following the decimal point. If the precision is not specified, a default precision of 6 is used.

Flags: -: Left aligns the output given the specified width, padding the output with spaces.  
+: Prints a preceding + sign for positive values. Negative numbers are always printed with the - sign.  
0: Pads the output with 0's when the formatted value has fewer characters than the width.  
space: Prints a preceding space for positive value.

## 