C CHEATSHEET

What is C?

C is a programming language developed at AT & T's Bell Laboratories of USA in 1972. It was designed and written by a man named Dennis Ritchie.

Boilerplate Code

```
#include<stdio.h>
int main(){
  return(0);
}
```

printf function

It is used to show output on the screen
`printf("Hello World!")`

scanf function
It is used to take input from the user
`scanf("placeholder", variables)`

Comments

A comment is the code that is not executed by the compiler, and the programmer uses it to keep track of the code.

Single line comment

```
// It's a single line comment
```

Multi-line comment

```
/*
  multi-line
  Comment
*/
```

Constants, Variables and Keywords

A constant is an entity that doesn't change, whereas, a variable is an entity that may change. A keyword is a word that carries special meaning.

Data types

Character type
A single character which has an ASCII value.
`char variable_name; // placeholder %c`

Integer type
The most natural size of integer for the machine.
`int variable_name; // placeholder %d`

Float type
A single-precision floating-point value.
`float variable_name; // placeholder %f`

Double type
A double-precision floating-point value.
`double variable_name; // placeholder %lf`

Void type
Represents the absence of the type
`void`

Escape Sequences

It is a sequence of characters starting with a backslash, and it doesn't represent itself when used inside string literal.

Backspace It adds a backspace \b
Newline Newline Character \n
Tab It gives a tab space \t
Backslash It adds a backslash \\

Conditional Instructions

If-else Statement

```
if (/* condition */)
{
/* code if condition is true */
}
else{
/* Code if condition is false */
}
```

if else-if Statement

```
if (condition) {
// Statements;
}
else if (condition){
// Statements;
}
else{
// Statements
}
```

Switch Case Statement

It allows a variable to be tested for equality against a list of values cases.

```
switch (expression)
{
case constant-expression:
statements;
 break;
default:
statement;
}
```

Arrays

An array is a collection of data items of the same type. They are denoted by [ ] or [size].

Declaration

int arr[] = {1, 2, 3, 4, 5}; // 5 integers are stored in arr

Accessing element

We access members of array using their position which usually starts from 0 to (size – 1).

Iterative Statements

Iterative statements execute any block of code lines repeatedly and can be controlled as per conditions added by the programmer.

while Loop

It allows execution of statement inside the block of the loop until the condition of loop succeeds.

```
while (/* condition */)
{
/* code */
}
```

do-while loop

It is an exit controlled loop. It is very similar to the while loop with one difference, i.e., the body of the do-while loop is executed at least once even if the expression is false

```
do
{
 /* code */
} while (/* condition */);
```

for loop

It is used to iterate the statements or a part of the program several times. It is frequently used to traverse the data structures like the array and linked list.

```
for (int i = 0; i < count; i++)
{
/* code */
}
```

Break Statement

break keyword inside the loop is used to terminate the loop

```
break;
```

Continue Statement

continue keyword skips the rest of the current iteration of the loop and returns to the starting point of the loop

```
continue;
```

Recursion

Recursion is a special function that calls (executes) itself multiple times. It is used to break a large problem into minor problems.

A base condition is essential to stop the recursion at some point, otherwise, the function runs infinitely.

Functions

Function is a piece of code that takes some input values and usually returns a single value after the operation.

Function Definition

```
return_type function_name(data_type parameter...){
  // code to be executed
  // return value
}
```

Calling Function

function_name (parameter...);

A function can be called either by value or by reference.

By Value

A copy of values are created. So, change in value doesn't affect the original value passed with the function call.

```
sum = calsum (a, b, c);
f = factr (a);
```

By Reference

The address of actual variables is passed. So, change is visible in the original value passed with the function call.

```
void swapNum (int *x, int *y)
```

Arrays

An array is a collection of data items of the same type. They are denoted by [ ] or [size].

Declaration

```
int arr[] = {1, 2, 3, 4, 5}; // 5 integers are stored in arr
```

Accessing element

We access members of array using their position which usually starts from 0 to (size – 1).

Strings

Strings are basically an array of characters terminated by a null ('\0') character. Include <strings.h> header file to use string functions

Declaring strings

```
char name[] = {'H', 'A', 'E', 'S', 'L', 'E', 'R', '\0'};
```

gets() function

It allows you to enter multi-word string

```
gets("string");
```

puts() function

It is used to show string output

```
puts("string");
```

strlen()

It is used to calculate the length of the string

```
strlen(string_name);
```

strcpy() function

It is used to copy the content of second-string into the first string passed to it

```
strcpy(destination, source);
```

strcat() function

It is used to concatenate two strings

```
strcat(first_string, second_string);
```

strcmp() function

It is used to compare two strings

```
strcmp(first_string, second_string);
```

Pointers

Pointers are variables which hold addresses of other variables.

Declaring pointers

```
int a = 4; // a is an integer with a value 4
int *p = &a; // p is a pointer holding the address of integer a
```

Accessing element stored in pointers

The * operator lets us access the value present at an address in memory with an intension of reading it or modifying it.

```
printf("value of a is %d", *p)
```

Structures

The structure is a collection of variables of different types under a single name. Defining structure means creating a new data type.

Structure syntax

```
struct structureName
{
  dataType member1;
  dataType member2;
  ...
};
```

typedef keyword

typedef function allows users to provide alternative names for the primitive and user-defined data types.

```
typedef struct structureName
{
  dataType member1;
  dataType member2;
  ...
}new_name;
```

The C Character Set

A character denotes any alphabet, digit or special symbol used to represent information.

Alphabets

A, B, ..., Y, Z a, b, ..., y, z

Digits

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Special symbols

~‘!@#%^&*()_-+=|\{} []:;"'<>,.?/$

File Handling

A set of methods for handling File I/O operation (read/write/append) in C language.

FILE pointer

FILE *filePointer;

Opening a file

It is used to open file in C.
filePointer = fopen(fileName.txt, w)

fscanf() function

It is used to read the content of file.
fscanf(FILE *stream, const char *format, ...)

fprintf() function

It is used to write content into the file.
fprintf(FILE *fptr, const char *str, ...);

fgetc() function

It reads a character from a file opened in read mode. It returns EOF on reaching the end of file.
fgetc(FILE *pointer);

fputc() function

It writes a character to a file opened in write mode

fputc(char, FILE *pointer);

Closing a file

It closes the file.
fclose(filePointer);