Engineering 150 Midterm Cheat Sheet Cheat Sheet

by Jeevan (sukablukagyat) via cheatography.com/201231/cs/42578/

Basic Operators

+	Addition
-	Subtraction
*	Multiplication
/	Division
=	Assign to Variable/Set equal to
%	Modulus (Int repersentation of remainder)
++var	increase before evaluation
var++	increase after evaluation
condition ? result : alternative;	short form of if-like structure

Input/Output

Cin	cin >> number;
Cout	cout << "Enter an integer: ";

Legal C++ Identifiers (The Rules for Var Names)

- Consists of letters (A-Z, Upper and Lowercase)
- Digits (NO DECIMALS)
- Underscores
- No Special Characters or Spaces
- No C++ Keywords by itself
- Must be Unique within it's namespace
- Case Sensitive

Example Switch Case

```
switch (operation) {
case 0:
// Things
break;
case 1:
// Things
break;
case 2:
                                    // Things
break;
```

Example Switch Case (cont)

```
>       default:
            // Things
            break;
}
```

Operation must be an Int, Char, or Boolean. Case: must reflect that datatype.

Example If Else

```
if ( conditional ) {
      // do something
}
else if ( anothe r_c ond itional ) {
      // do something else
}
else {
      // do something as default
}
```

Example For Sentence

```
for ( init; test; command ) {
      // do something
}
```

"break;" and "continue;" identical effects.

Example Do While Sentence

```
do {
      //do something
} while (bool expres  sion);
```

Pointers

int *ptr;	Pointer definition
ptr = &var1;	ptr set to address of var1
var2 = *ptr;	set var2 to value of var1

Struct Example

```
// Declare Structure
struct Order {
       int modelN  umber;
        string phone;
        float cost;
};
// Main Function, define the structure variable
int main( ) {
        Order item1;
       // Give values to structure variables
        ite m 1.m  od  el N umber = 10;
        ite m1.p hone = "  Sa msung Galaxy S";
        ite m1.cost = 1099.99;
       // Displa  y/ O utput structure members
       cout << "  Order, " << item1.p  hone <<
item1.m  o d e lN  umber;
       cout << "This item costs: $" << part1.cost
<< endl;
}
// Output:
// Order, Samsung Galaxy S10
// This item costs: $1099.99
```

Boolean Logic

==	Test of Equality
!=	Test of Non-Equality
<	Less Than
>	Greater Than
<=	Less Than or Equal
>=	Greater Than or Equal
!	NOT
&&	AND
||	OR

Boolean expressions in C++ are evaluated left-to-right!

Data Types

int	3
float	3.0f
double	3.0
char	'a'
string	"Hello World!"
bool	true/false

File Input / Output

```
#include <fstream.h>
ifstream file; //read buffer
ofstream file; //write buffer
file.open ("fi len ame ", [file mode consta nt]);
  //Test if the file was created
if(fs.i s_ open()) if(fs)
//Read s/W rites like cin and cout
file >> var; //Read
file << ''Text: " << var << endl; //Write
//Read Entire line
getline (file, Str ing);
//Read until it arrives at the end of file
while( fil e.e of())
//Detect if the read/write fail
if(fil e.f ail())
//Close File
file.c lose();
```

Procedures

```
//Declaration
 void Proced ure Name()
{
     // do something
}
//Call to procedure
Proced ure Name();
```

In the procedures we don't receive variables and don't return other variable.

Functions

```
//Declaration
  [re tur nType] functi onName ( [input 1Type
input1 Name, input2Type input2 Name, ....] )
{
     // do something
      return value; // value must be of type
returnType
}
//Call to function
[retur ntype var =] functi onName ([inpu t1Type
input1 Name, input2Type input2 Name, ....])
```

We have two methods to create and call functions:
passed with values and passed for reference.

Pass by reference: we put & before variable in the declaration.
ReturnType must be a Datatype
Cannot return more than one thing

Dynamic Array Template

```
dataType *nameOfTheDynamicArray;
nameOf The Dyn ami cArray = new dataTy pe[ num -
ofe lem ents];
nameOf The Dyn ami cAr ray[0] = value;
delete [] nameOf The Dyn ami cArray;
```

"new" function allows dynamic array to change it size, since they are parameters.

"delete" will delete everything stored in the dynamic array.

enum example

```
enum gasType {gas, diesel};
gasType gasT;
```