

**Include Headers**

```
#include <headerfile>
```

**Common Headers**

```
iostream, fstream, math, ctype, string
```

**Namespace**

```
using namespace std;
```

**Data Types**

```
int, char, float, double, void, bool
```

**Comments**

```
// Comment text
```

```
/* Multi-line comment text */
```

**Arithmetic Operators**

```
+ (Addition), - (Subtraction), * (Multiplication), / (Division), % (Modulus)
```

**Relational Operators**

```
< (Less Than), <= (Less Than or Equal To), > (Greater Than),  
>= (Greater Than or Equal To), == (Equal To), != (Not Equal To)
```

**Logical Operators**

```
|| (logical OR), && (logical AND), ! (logical NOT)
```

**Pointers**

```
int *ptr; //Define pointer
```

```
ptr = &var //ptr set to address of var
```

```
var2 = *ptr //Set var2, to value of var1
```

**If Else**

```
if(<condition>)
```

```
{ <statement 1>; }
```

```
else
```

```
{ <statement 2>; }
```

**For Loop**

```
for(<initialize>;<condition>;<update>)
```

```
{ <statement>; }
```

**While Loop**

```
while (<condition>)
```

```
{ <statement>; }
```

**Do-While Loop**

```
do { <statement>; }
```

```
while (<condition>;)
```

**Switch Statement**

```
switch(<expression> )
```

```
{
```

```
case <constant1>:
```

```
<statement sequence 1>;
```

```
break;
```

```
case <constant2>:
```

```
<statement sequence 2>;
```

```
break;
```

```
case <constantn+1>:
```

```
<statement sequence n+1>;
```

```
break;
```

```
[ default:
```

```
<statement sequence n>;
```

```
break;]
```

```
}
```

**Arrays**

```
//New 5 element array
```

```
int myArray[5];
```

```
//Array index starts at 0
```

```
//Access 3rd Element
```

```
myArray[2]=var;
```

**I/O Operators**

```
>> //Input Operator
```

```
<< //Output Operator
```

```
cin >> var1, var2, var3;
```

```
cout << "TEXT: " << var1 << endl;
```

```
cin.get(char* buffer, streamsize num, char delim );
```

**File I/O**

```
fstream file;
```

```
file.open("filename",<file mode constant>);
```

```
//Reads and Writes like cin and cout
```

```
file >> var;
```

```
file << "Text: " << var << endl;
```

```
// Read Entire Line
```

```
getline (file,line);
```

```
//Reading Writing Binary Data
```

```
file.read(memory_block, size);
```

```
file.write(memory_block, size);
```

```
file.close();
```

**File Mode Constants**

```
ios::in //Opens file for reading
```

```
ios::out //Opens file for writing
```

```
ios::ate //Seeks the EOF.I/O operations can occur anywhere
```

```
ios::app //Causes output to be appended at EOF
```

```
ios::trunc //Destroys the previous contents
```

```
ios::nocreate //Causes open() to fail if file doesnt already exist
```

```
ios::noreplace //Causes open() to fail if file already exists
```

**Function Prototype**

```
<return_data_type> <function_name> (parameter list)
```

```
{ body of the function }
```

**Class Prototype**

```
class <class_name>
```

```
{
```

```
public:
```

```
//method_prototypes
```

```
protected:
```

```
//method_prototypes
```

```
private:
```

```
//method_prototypes
```

```
//data_attributes
```

```
};
```

**Structure Prototype**

```
struct <structure_name> {
```

```
member_type1 member_name1;
```

```
member_type2 member_name2;
```

```
} <object_name>;
```

**Accessing Data Structures**

```
//Access member variable from Struct/Class
```

```
myStruct.membervar1 = var;
```

```
//Call Class Method
```

```
myClass.method1(args);
```

```
//Pointer to Struct/Class
```

```
myStructType *ptr;
```

```
ptr = &myStruct;
```

```
ptr->membervar1 = var;
```