How to use GDB: Basic Commands

- **Compiling your program**

All source files must be compiled with the -g flag. For example: *g++ -g -c main.cc*

- **Running GDB**

To run gdb, type: *gdb <executable file name>*

- **To exit from GDB**

```
q
```

- **To run your program** (possibly after setting some breakpoints)

```
r
```

- **To run your program with command-line args**

r *arg1 arg2 ...*

- **To look at source code**

l <fn name>	Print 10 lines, centered around the start of the given function.
l	Print 10 more lines.
l -	
l <start #>,<stop #>	Print lines <start #> through <stop #>.

- **Breakpoints**

b <fn name>	Stop at entry to the given function.
b <line #>	Stop at the given line of the current file.
info b	To see what breakpoints are set.
clear <fn name>	Remove breakpoint at entry to given function.
clear <line #>	Remove breakpoint at given line of current file.
delete <breakpoint #>	Remove a single breakpoint (use ‘info b’ to find breakpoint num
cond <#> <cond>	Stop at breakpoint <#> only if condition <cond> evaluates t <cond> can be any C++ expression.
commands	Use this after setting a breakpoint or after stopping at a breakpo specify *gdb* commands that are to be executed every time exe stops at this breakpoint. You will be asked to type command per line, ending with "end".
c	Continue execution after stopping at a breakpoint.
s	
n	Like ‘s’, but execute function calls as a single unit.

- **To look at and/or change the values of variables**

```
p <exp>
```

Print the value of the given expression. The expression can be any legal C++ expression, including a function call, *e.g.*, L.myItems[0], L.CurrentItem(), *etc*.
T}

```
set <variable>=<exp>
```

- **Call information**

```
bt
```

- **Help information**

```
help
```

- **C++ classes and class templates**

To refer to a class member function (*e.g.* to look at the source code or to set a breakpoint) use: *<class name>::<function name>*. For example: *b StringList::CurrentItem*

To refer to a member function of a class derived from a class template use: *’<class template name><<type>>::<function name>(<args>)’*. For example: *b ’List<String>::CurrentItem(void)’*