Development > Programming Languages > C++

The C++ 20 Masterclass: From Fundamentals to Advanced

Learn and Master Modern C++ From Beginning to Advanced in Plain English: C++11, C++14, C++17, C++20 and More!

4.7 ★★★★☆

Created by Daniel Gakwaya

Slides

Section: Function call stack and debugging

Slide intentionally left empty

Function call stack & Debugging

```
int main(int argc, char **argv){
   int a{10};
   int b{12};
   int summation = sum(a,b);
    std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
   int adjustment{2};
   input += adjustment;
   return input;
int sum(int x , int y){
   int result = x + y;
   adjust(result);
   return result;
```

main dvanced © Dan

ret_v params address

x 10
y 12
result = x + y = 22

ret_v params address

a 10
b 12

ret_summation xxx



- CodeLite
- Microsoft Visual Studio

Locals Watches Call Stack Slide intentionally left empty

Function call stack

```
#include <iostream>
int add_numbers(int a, int b)
    return a + b;
int main()
    int a = 10;
    int b = 5;
    int c;
    std::cout << "Statement1" << std::endl;</pre>
    std::cout << "Statement2" << std::endl;</pre>
    c = add_numbers(a, b);
    std::cout << "Statement3" << std::endl;</pre>
    std::cout << "Statement4" << std::endl;</pre>
    return 0;
```



Compiler

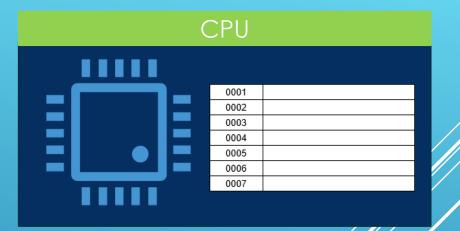






```
a = 10 (int)
b = 5 (int)
c (int)
print("Statement1")
print("Statement2")
c = f_add(a,b)
print("Statement3")
print("Statement4")
end
```

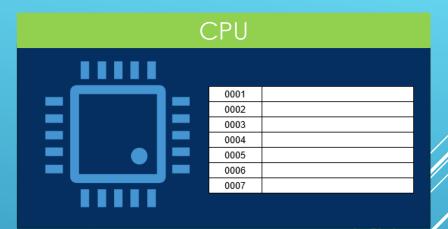
					1
	0001				
	0002				
	0003				
	0004				
Program	0005				
area	0006				
arca	0007				
	8000				
	0009				
	0010				
	0020				
+ 20 Masterclass : From I	0030 undamer	tals to Advo	anced (© Daniel	Gakw





a = 10 (int)
b = 5 (int)
c (int)
<pre>print("Statement1")</pre>
<pre>print("Statement2")</pre>
c = f_add(a,b)
<pre>print("Statement3")</pre>
<pre>print("Statement4")</pre>
end 10

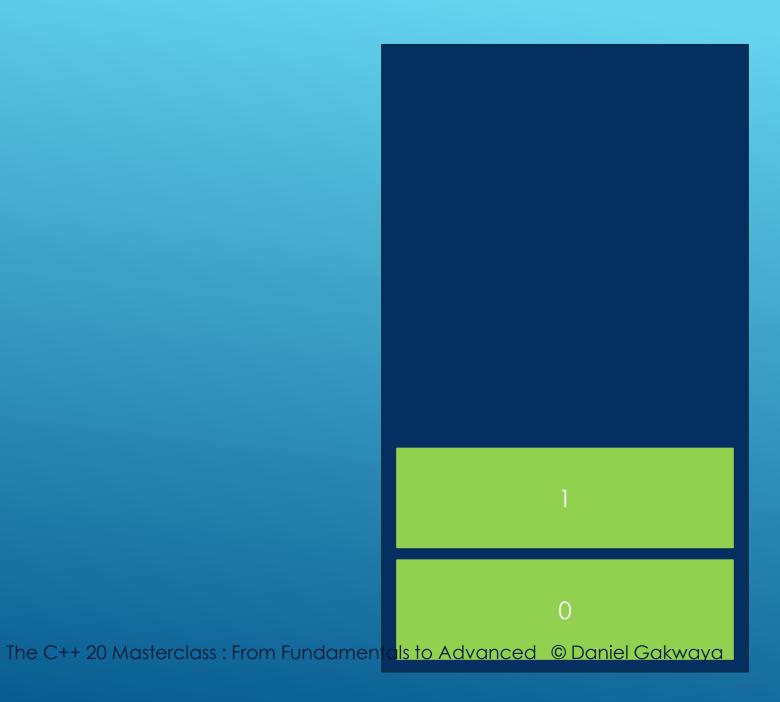


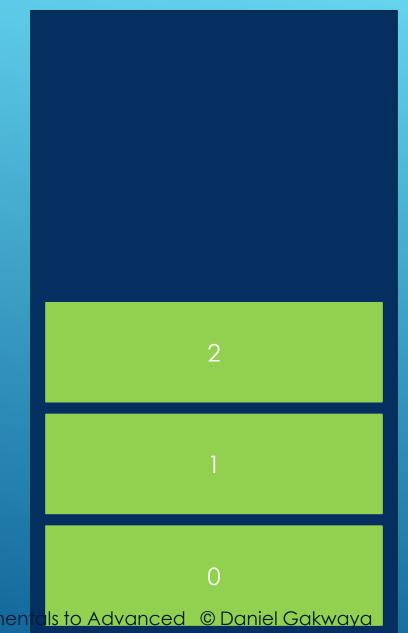


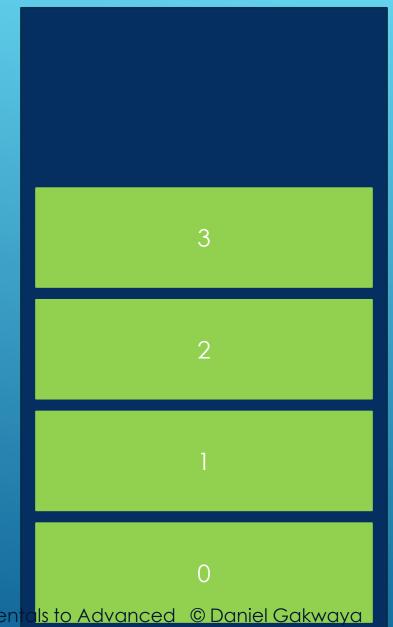


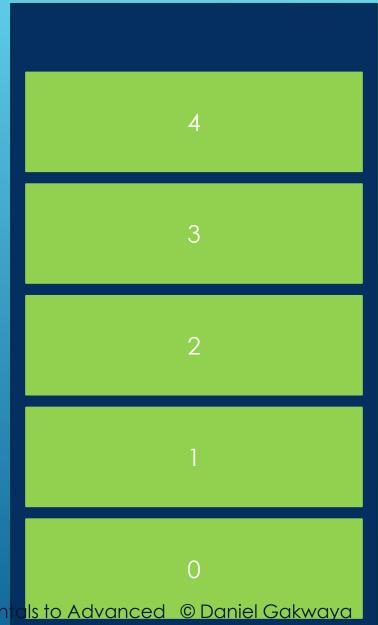
```
a = 10 (int)
b = 5 (int)
c (int)
print("Statement1")
print("Statement2")
c = f_add(a,b)
print("Statement3")
print("Statement4")
end
```



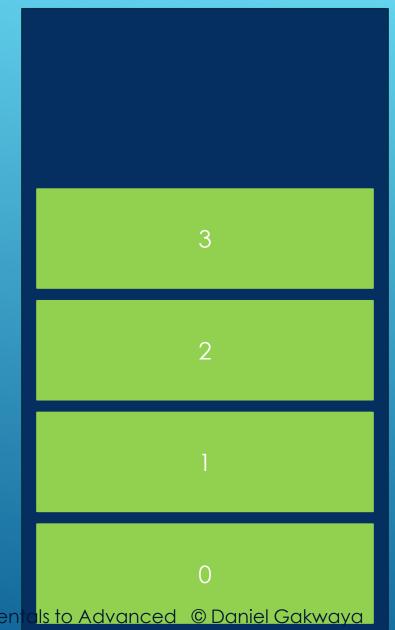


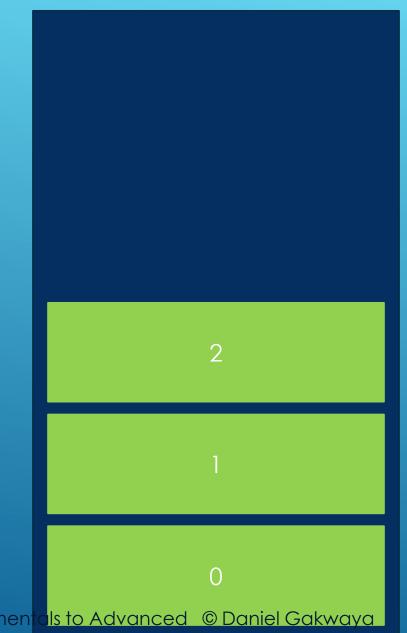


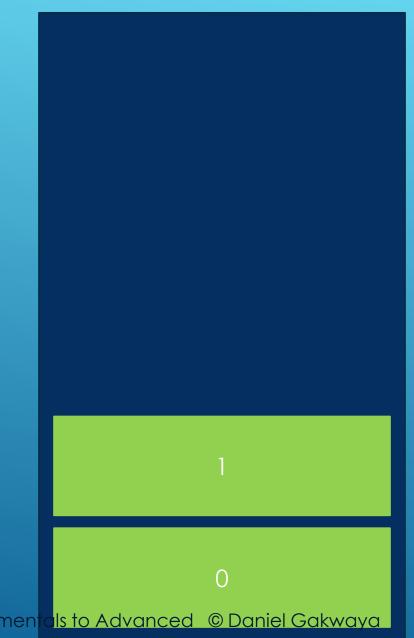




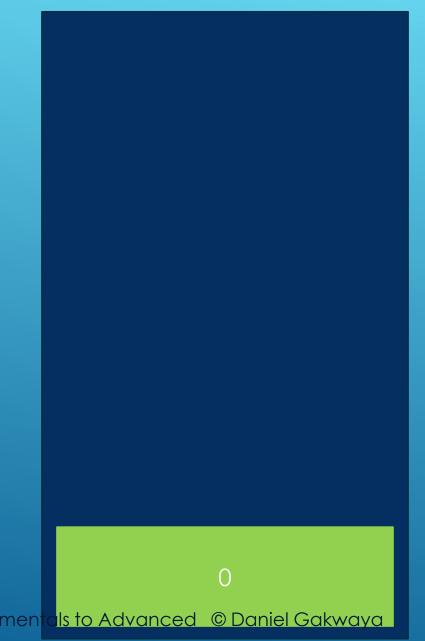
16



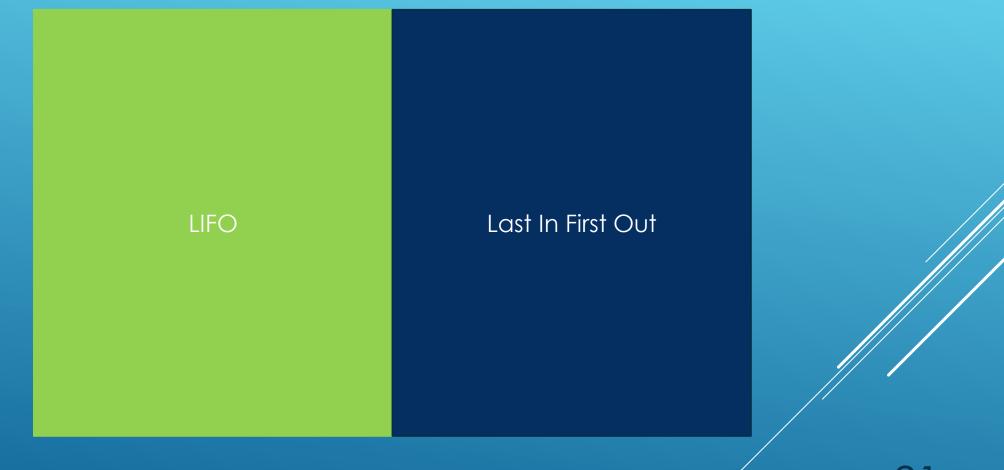












```
int main(int argc, char **argv){
    int a{10};
    int b{12};
    int summation = sum(a,b);
    std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
    int adjustment{2};
    input += adjustment;
    return input;
int sum(int x , int y){
    int result = x + y;
    adjust(result);
    return result;
```

```
int main(int argc, char **argv){
    int a{10};
    int b{12};
    int summation = sum(a,b);
    std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
    int adjustment{2};
    input += adjustment;
    return input;
int sum(int x , int y){
    int result = x + y;
    adjust(result);
    return result;
```

a 10 b 12 ercummation xxx

main

```
int main(int argc, char **argv){
    int a{10};
    int b{12};
    int summation = sum( a , b);
    std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
    int adjustment{2};
    input += adjustment;
    return input;
int sum(int x , int y){
    int result = x + y;
    adjust(result);
    return result;
```

```
ret_v params address

a 10
b 12
el Garwayation xxx
```

main .dvanced © D

```
int main(int argc, char **argv){
   int a{10};
   int b{12};
   int summation = sum( a , b);
    std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
   int adjustment{2};
   input += adjustment;
   return input;
int sum(int x , int y){
   int result = x + y;
   adjust(result);
   return result;
```

result ret_v

params

main

The C++ 20 Masterclass: From Fundamentals to Advanced © Daniel Gdkwayation XXX

a

b

address

```
int main(int argc, char **argv){
   int a{10};
   int b{12};
   int summation = sum(a,b);
    std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
   int adjustment{2};
   input += adjustment;
   return input;
int sum(int x , int y){
   int result = x + y;
   adjust(result);
   return result;
```

main

c

y 12
result = x + y = 22

ret_v params address

a 10
b 12
else way ation xxx

```
int main(int argc, char **argv){
   int a{10};
   int b{12};
   int summation = sum(a,b);
    std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
   int adjustment{2};
   input += adjustment;
   return input;
int sum(int x , int y){
   int result = x + y;
   adjust(result);
   return result;
```

main

address ret_v params result = x + y = 22ret_v address params b The C++ 20 Masterclass: From Fundamentals to Advanced © Daniel Gdkwayation XXX

```
int main(int argc, char **argv){
   int a{10};
   int b{12};
   int summation = sum(a,b);
   std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
   int adjustment{2};
   input += adjustment;
   return input;
int sum(int x , int y){
   int result = x + y;
   adjust(result);
   return result;
```

sum

main

adjustment input

address ret_v params

result = x + y = 22

ret_v address params

a

b

XXX

```
int main(int argc, char **argv){
   int a{10};
   int b{12};
   int summation = sum(a,b);
   std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
   int adjustment{2};
   input += adjustment;
   return input;
int sum(int x , int y){
   int result = x + y;
   adjust(result);
   return result;
```

sum

main

adjustment Input (ref_result)

address ret_v params

result = x + y = 22

ret_v address params

a

b

XXX

```
int main(int argc, char **argv){
   int a{10};
   int b{12};
   int summation = sum(a,b);
   std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
   int adjustment{2};
   input += adjustment;
   return input;
int sum(int x , int y){
   int result = x + y;
   adjust(result);
   return result;
```

sum

main dvanced © Dan adjustment
Input (ref_result)

ret_v params address

x 10

result = x + y = 22

ret_v params address

XXX

a 10

b 12

30

```
int main(int argc, char **argv){
   int a{10};
   int b{12};
   int summation = sum(a,b);
   std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
   int adjustment{2};
   input += adjustment;
   return input;
int sum(int x , int y){
   int result = x + y;
   adjust(result);
   return result;
```

sum

main vanced © Dan adjustment
Input (ref_result)

ret_v params address

x 10
y 12
result = 24

ret_v params address

a 10
b 12
commodion xxx

```
int main(int argc, char **argv){
   int a{10};
   int b{12};
   int summation = sum( a , b);
    std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
   int adjustment{2};
   input += adjustment;
   return input;
int sum(int x , int y){
   int result = x + y;
   adjust(result);
   return result;
```

sum

main

```
address
                                                        ret_v
                                                                     params
                                                         result = 24
                                                        ret_v
                                                                                  address
                                                                     params
                                                         b
The C++ 20 Masterclass: From Fundamentals to Advanced © Daniel Gdkwayation
                                                                       XXX
```

```
int main(int argc, char **argv){
   int a{10};
   int b{12};
   int summation = sum( a , b);
    std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
   int adjustment{2};
   input += adjustment;
   return input;
int sum(int x , int y){
   int result = x + y;
   adjust(result);
   return result;
```

sum

result = 24

main

ret_v address params b XXX

```
int main(int argc, char **argv){
   int a{10};
   int b{12};
   int summation = sum( a , b);
    std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
   int adjustment{2};
   input += adjustment;
   return input;
int sum(int x , int y){
   int result = x + y;
   adjust(result);
   return result;
```

main

ret_v params address
a 10
b 12
er surwayation xxx

```
int main(int argc, char **argv){
    int a{10};
    int b{12};
    int summation = sum(a,b);
    std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
    int adjustment{2};
    input += adjustment;
    return input;
int sum(int x , int y){
    int result = x + y;
    adjust(result);
    return result;
```

a 10 b 12 Jawwation xxx

main

```
int main(int argc, char **argv){
    int a{10};
    int b{12};
    int summation = sum(a,b);
    std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
    int adjustment{2};
    input += adjustment;
    return input;
int sum(int x , int y){
    int result = x + y;
    adjust(result);
    return result;
```

b 12 ergammyation 24

main

Understanding the call stack: some benefits

- Truly understanding how function local variables become alive and die
- The need for inline functions becomes obvious
- Understanding the debug process of your program becomes easier. We are going to learn more about that in the next lecture.

Debugging

Debugging

Running your program through some other program to make it freeze at some point. This gives the ability to execute it line by line, jumping into functions, and examining the local variables in the current stack activation record.

```
#include <iostream>
int add_numbers(int a, int b)
    return a + b;
int main()
    int a = 10;
    int b = 5;
    int c;
    std::cout << "Statement1" << std::endl;</pre>
    std::cout << "Statement2" << std::endl;</pre>
    c = add_numbers(a, b);
    std::cout << "Statement3" << std::endl;</pre>
    std::cout << "Statement4" << std::endl;</pre>
    return 0;
```



Compiler

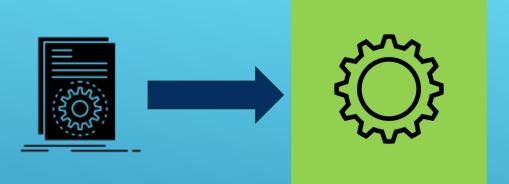






```
a = 10 (int)
b = 5 (int)
c (int)
print("Statement1")
print("Statement2")
c = f_add(a,b)
print("Statement3")
print("Statement4")
end
```





```
Developer PowerShell for VS 2019 Preview

*******************

** Visual Studio 2019 Developer PowerShell v16.9

** Copyright (c) 2020 Microsoft Corporation

******************

PS C:\Users\Gakwaya\source\repos>
```

```
■int main(int argc, char **argv){
 68
 69
          int a{10};
 70
          int b{12};
          int summation = sum(a,b);
          std::cout << "sum : "</pre>
 74
               << summation << std::endl;
 76
          return 0;
 77
 78
     ■int& adjust( int& input){
 79
 80
          int adjustment{2};
 81
          input += adjustment;
 82
          return input;
 83
 84
     ■int sum(int x , int y){
 85
 86
 87
          int result = x + y;
88 adjust(result);
ragg Fundameptals to Adygnced © Daniel Gakwaya
```

```
OnlineCourses\9.CppMasterClass\DemoCodeV2\22.FunctionCallStackAndDebugging\22-03Debugging\main.cpp
                    d Debugger Plugins Perspective Settings PHP Help
                      22-01Introduction\main.cpp × Next 02FunctionCallStack\main.cpp ×
                                                                     22-03Debugging\main.cpp
                                  int summation = sum(a,b);
                                  std::cout << "sum : "</pre>
                      73
                                       << summation << std::endl;
                      74
                      75
                      76
                                  return 0;
                      77
                      78
                           ■int& adjust( int& input){
                      80
                                  int adjustment{2};
                                  input += adjustment;
                      82
                                  return input;
                      83
                      84
                           \blacksquare int sum(int x , int y){
                      86
                      87
                                  int result = x + y;
                                  adjust(result);
                      88
                                  return result:
The C++ 20 Masterclass: From Fundamentals
```

44

Locals Watches Call Stack



- CodeLite
- Microsoft Visual Studio

Debugging in Visual Studio Code

```
int main(int argc, char **argv){
    int a{10};
    int b{12};
    int summation = sum(a,b);
    std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
    int adjustment{2};
    input += adjustment;
    return input;
int sum(int x , int y){
    int result = x + y;
    adjust(result);
    return result;
```

Locals Watches Call Stack



Run/Debug configurations in launch.json files

Debug tools from gcc

```
"name": "g++.exe - Build and debug active file",
 9
                "type": "cppdbg",
10
                "request": "launch",
11
                "program": "${fileDirname}\\rooster.exe",
12
13
                "args": [],
                "stopAtEntry": false,
14
                "cwd": "${workspaceFolder}",
15
                "environment": [],
16
17
                "externalConsole": true,
                "MIMode": "gdb",
18
                "miDebuggerPath": "C:\\mingw64\\bin\\gdb.exe",
19
                "setupCommands": [
20
21
                        "description": "Enable pretty-printing for gdb",
22
                        "text": "-enable-pretty-printing",
23
                        "ignoreFailures": true
24
25
26
                "preLaunchTask": "Build with GCC 10.0.2 Mingw"
27
28
            },
```

Debug tools from the Microsoft C++ compiler

```
29
                "name": "cl.exe build and debug active file",
30
                "type": "cppvsdbg",
31
                "request": "launch",
32
                "program": "${fileDirname}\\rooster.exe",
33
                "args": [],
34
35
                "stopAtEntry": false,
                "cwd": "${workspaceFolder}",
36
                "environment": [],
37
38
               "console": "externalTerminal",
                "preLaunchTask": "Build with MSVC"
39
40
```

Debugging In CodeLite

Debugging

Running your program through some other program to make it freeze at some point. This gives the ability to execute it line by line, jumping into functions, and examining the local variables in the current stack activation record.

```
22-01Introduction\main.cpp × 22-02FunctionCallStack\main.cpp × 22-03Debugging\main.cpp
                                           ■int main(int argc, char **argv){
                                       67
                                       68
                                                 int a{10};
                                       69
                                       70
                                                 int b{12};
                                       71
                                       72
                                                 int summation = sum(a,b);
                                                 std::cout << "sum : "</pre>
                                       73
                                       74
                                                      << summation << std::endl;
                                       75
                                       76
                                                 return 0;
                                       77
               Break points
                                       78
                                       79
                                           ■int& adjust( int& input){
                                       80
                                                 int adjustment{2};
                                                 input += adjustment;
                                       81
                                       82
                                                 return input;
                                       83
                                       84
                                       85
                                           ■int sum(int x , int y){
                                       86
                                       87
                                                 int result = x + y;
                                                 adjust(result);
                                       88
                                                 return result;
                                       89
The C++ 20 Masterclass: From Fundamentals to Advanced © Daniel Gakwaya
```

```
OnlineCourses\9.CppMasterClass\DemoCodeV2\22.FunctionCallStackAndDebugging\22-03Debugging\main.cpp
                    d Debugger Plugins Perspective Settings PHP Help
                      22-01Introduction\main.cpp × Next 02FunctionCallStack\main.cpp ×
                                                                     22-03Debugging\main.cpp
                                  int summation = sum(a,b);
                                  std::cout << "sum : "</pre>
                      73
                                       << summation << std::endl;
                      74
                      75
                      76
                                  return 0;
                      77
                      78
                           ■int& adjust( int& input){
                      80
                                  int adjustment{2};
                                  input += adjustment;
                      82
                                  return input;
                      83
                      84
                           \blacksquare int sum(int x , int y){
                      86
                      87
                                  int result = x + y;
                                  adjust(result);
                      88
                                  return result:
The C++ 20 Masterclass: From Fundamentals
```

Locals Watches Call Stack

Debugging in Microsoft Visual Studio

Debugging

Running your program through some other program to make it freeze at some point. This gives the ability to execute it line by line, jumping into functions, and examining the local variables in the current stack activation record.

```
22-01Introduction\main.cpp × 22-02FunctionCallStack\main.cpp × 22-03Debugging\main.cpp
                                            ■int main(int argc, char **argv){
                                        67
                                        68
                                                  int a{10};
                                        69
                                        70
                                                  int b{12};
                                        71
                                        72
                                                  int summation = sum(a,b);
                                                  std::cout << "sum : "</pre>
                                        73
                                        74
                                                       << summation << std::endl;
                                        75
                                        76
                                                  return 0;
                                        77
               Break points
                                        78
                                        79
                                            ■int& adjust( int& input){
                                        80
                                                  int adjustment{2};
                                                  input += adjustment;
                                        81
                                        82
                                                  return input;
                                        83
                                        84
                                        85
                                            \blacksquare int sum(int x , int y){
                                        86
                                        87
                                                  int result = x + y;
                                                   adjust(result);
                                        88
                                                  return result;
                                        89
The C++ 20 Masterclass: From Fundamentals to Advanced © Daniel Gakwaya
```

```
OnlineCourses\9.CppMasterClass\DemoCodeV2\22.FunctionCallStackAndDebugging\22-03Debugging\main.cpp
                    d Debugger Plugins Perspective Settings PHP Help
                      22-01Introduction\main.cpp × Next 02FunctionCallStack\main.cpp ×
                                                                     22-03Debugging\main.cpp
                                  int summation = sum(a,b);
                                  std::cout << "sum : "</pre>
                      73
                                       << summation << std::endl;
                      74
                      75
                      76
                                  return 0;
                      77
                      78
                           ■int& adjust( int& input){
                      80
                                  int adjustment{2};
                                  input += adjustment;
                      82
                                  return input;
                      83
                      84
                           \blacksquare int sum(int x , int y){
                      86
                      87
                                  int result = x + y;
                                  adjust(result);
                      88
                                  return result:
The C++ 20 Masterclass: From Fundamentals
```

Locals Watches Call Stack

Debugging arrays, loops and pointers

```
#include <iostream>
double sum ( double array[], size_t count){
    double sum{};
    for(size_t i{} ; i < count ; ++i){</pre>
        sum += array[i];
    return sum;
int main(int argc, char **argv)
    double numbers[] {10.0,20.0,30.0,40.0,50.0}; // Sum should be 150.0
    double total = sum(numbers, std::size(numbers));
    std::cout << "sum : " << total << std::endl;</pre>
    return 0;
```

Function call stack & Debugging : Summary

```
int main(int argc, char **argv){
   int a{10};
   int b{12};
   int summation = sum(a,b);
    std::cout << "sum : "</pre>
        << summation << std::endl;
    return 0;
int& adjust( int& input){
   int adjustment{2};
   input += adjustment;
   return input;
int sum(int x , int y){
   int result = x + y;
   adjust(result);
   return result;
```

sum

main Ivanced © Dan

```
ret_v params address

x 10
y 12
result = x + y = 22

ret_v params address

a 10
b 12

iel sdrwwyation xxx
```

72



- CodeLite
- Microsoft Visual Studio