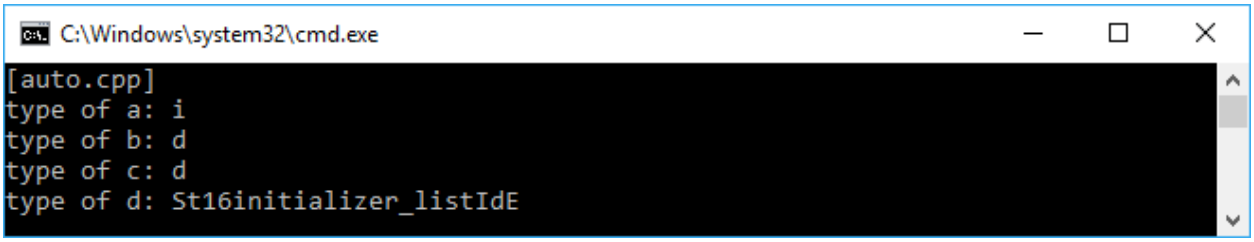
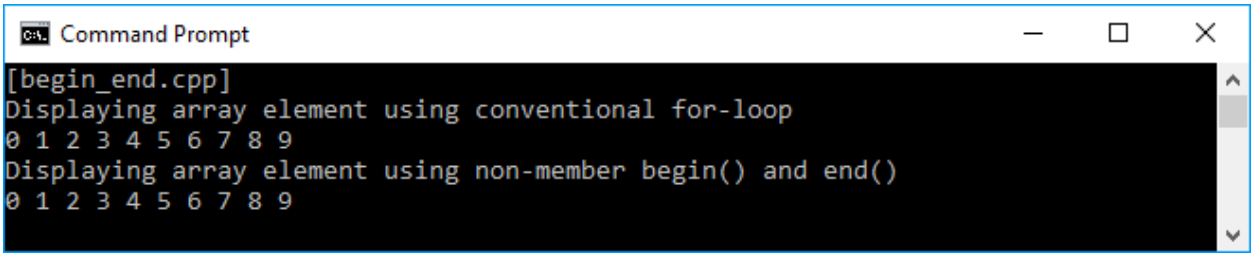
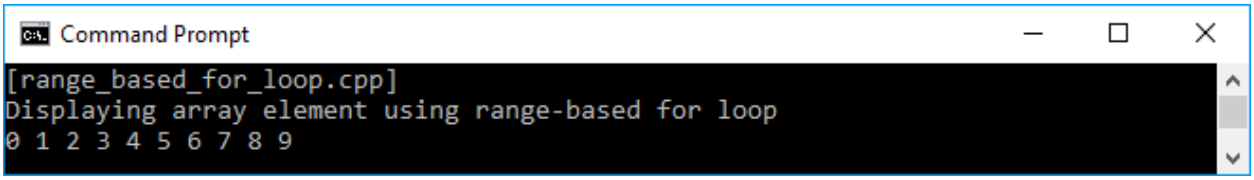
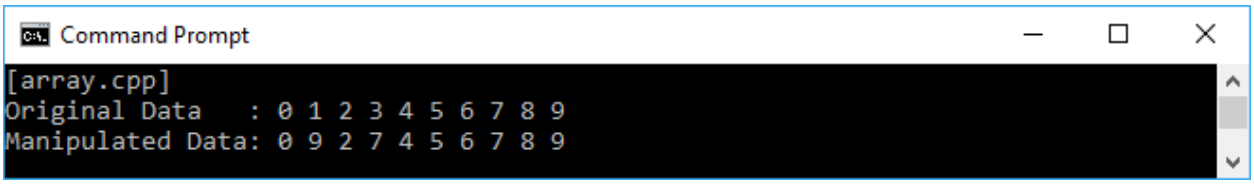
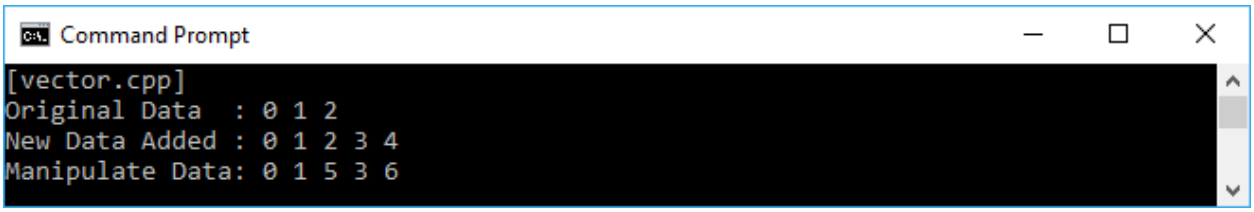
Chapter 1: Diving into Modern C++

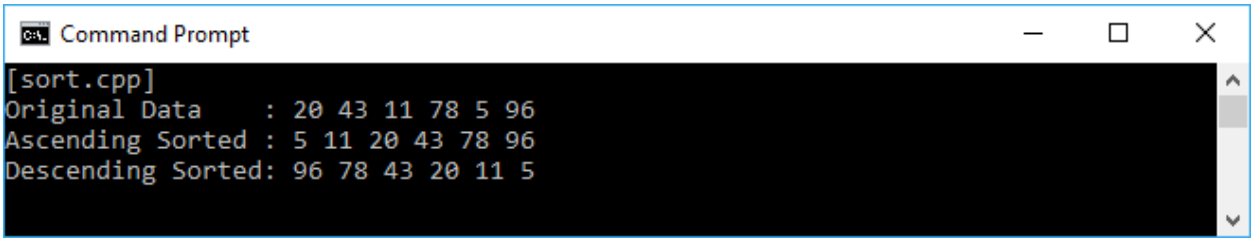


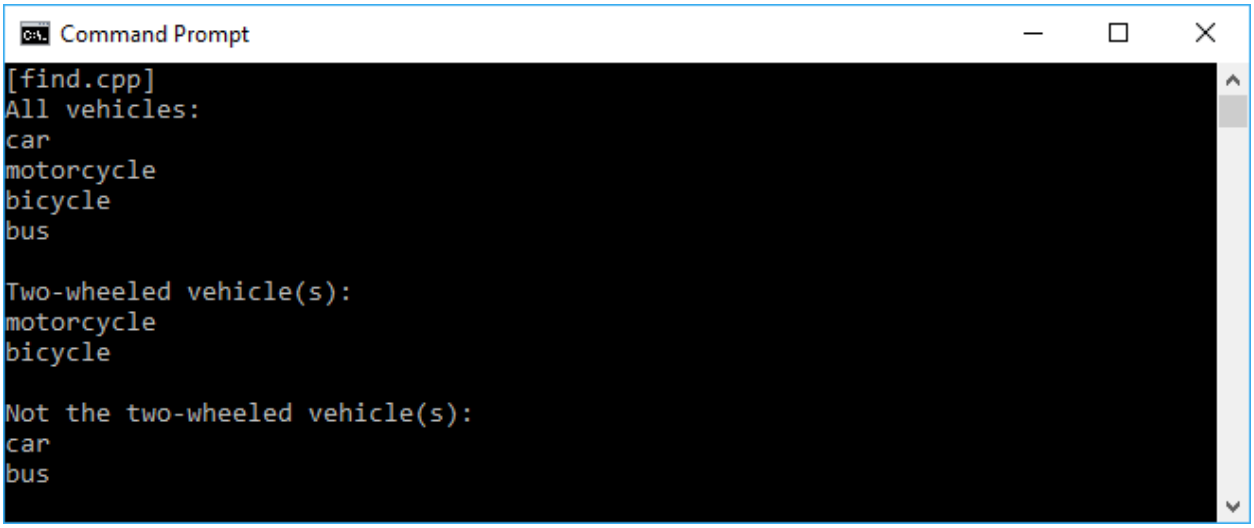


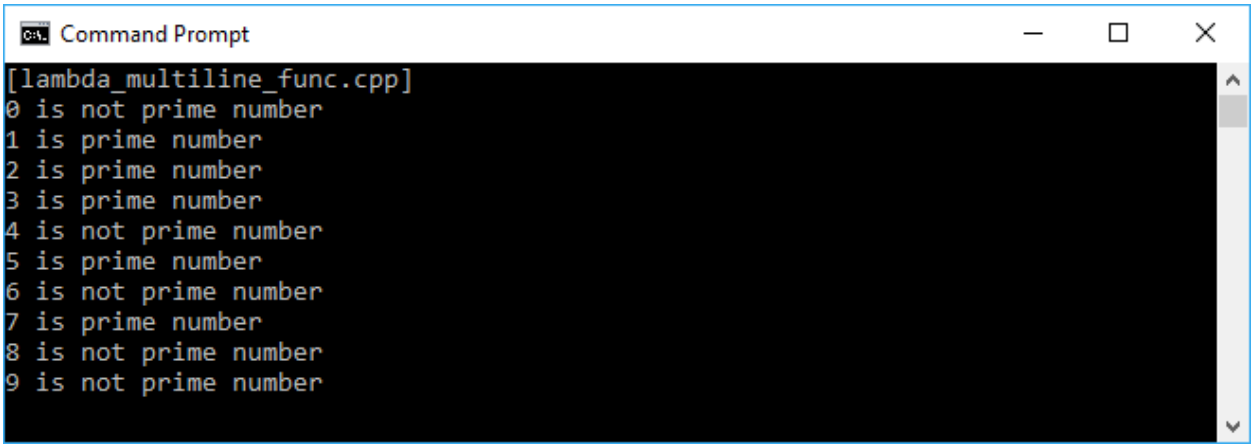

Command Prompt
[range_based_for_loop.cpp]
Displaying array element using range-based for loop
0 1 2 3 4 5 6 7 8 9

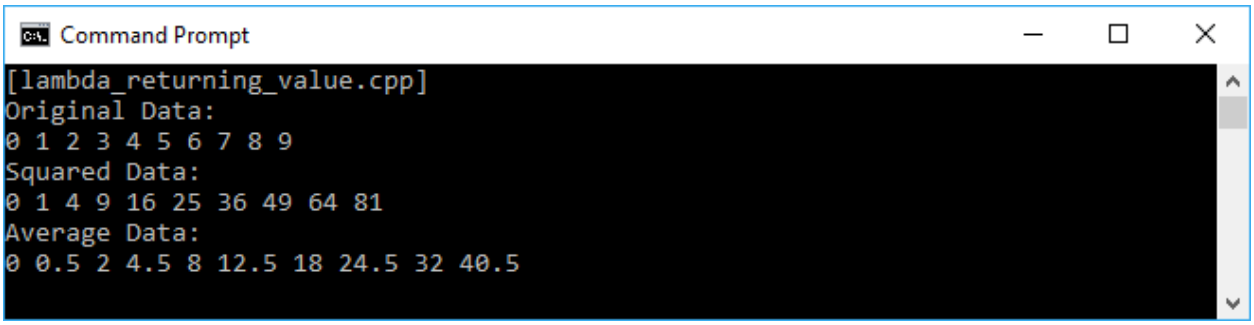

Command Prompt
[array.cpp]
Original Data   : 0 1 2 3 4 5 6 7 8 9
Manipulated Data: 0 9 2 7 4 5 6 7 8 9

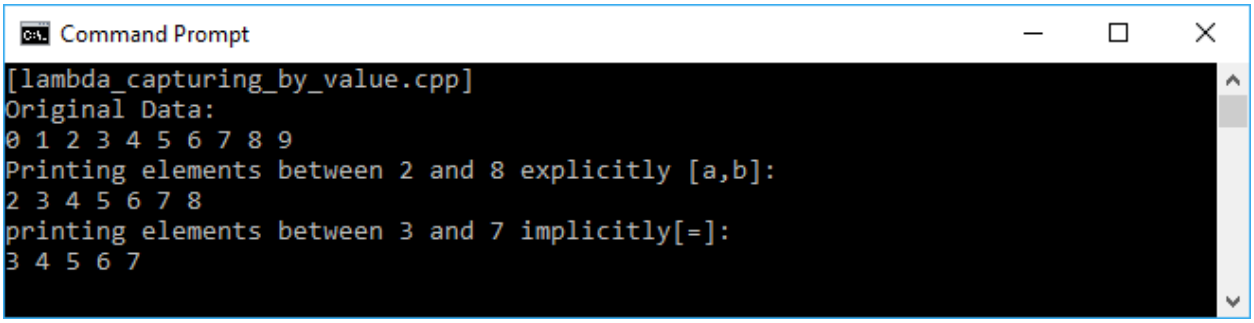

Command Prompt
[vector.cpp]
Original Data : 0 1 2
New Data Added : 0 1 2 3 4
Manipulate Data: 0 1 5 3 6

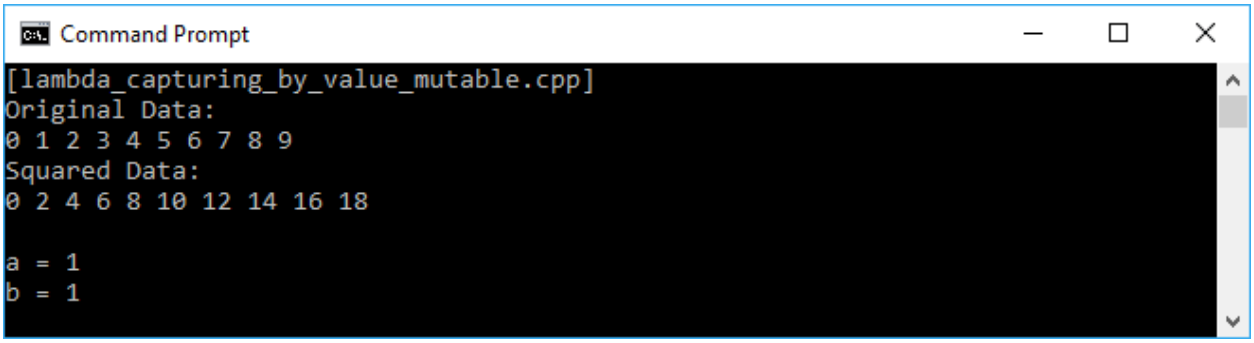

Command Prompt
[sort.cpp]
Original Data   : 20 43 11 78 5 96
Ascending Sorted : 5 11 20 43 78 96
Descending Sorted: 96 78 43 20 11 5

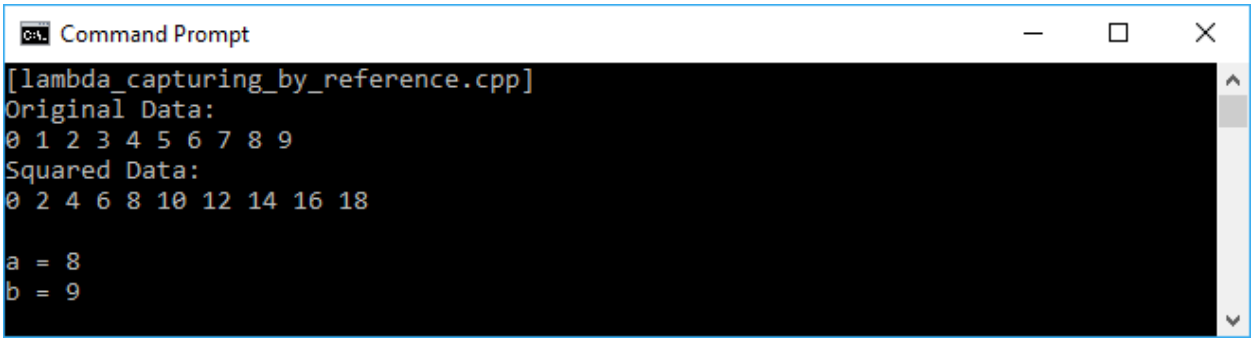

Command Prompt
[find.cpp]
All vehicles:
car
motorcycle
bicycle
bus
Two-wheeled vehicle(s):
motorcycle
bicycle
Not the two-wheeled vehicle(s):
car
bus

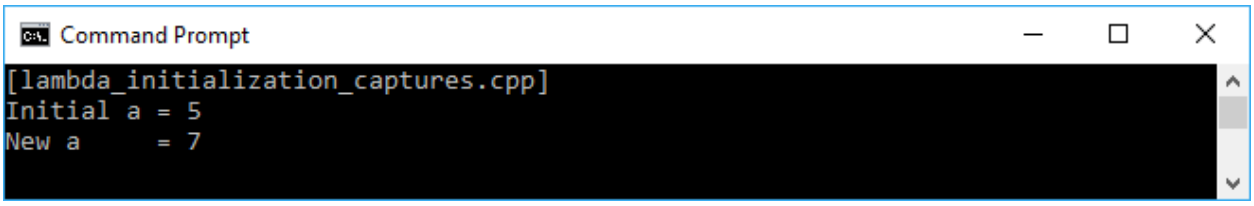

Command Prompt
[lambda_multiline_func.cpp]
0 is not prime number
1 is prime number
2 is prime number
3 is prime number
4 is not prime number
5 is prime number
6 is not prime number
7 is prime number
8 is not prime number
9 is not prime number

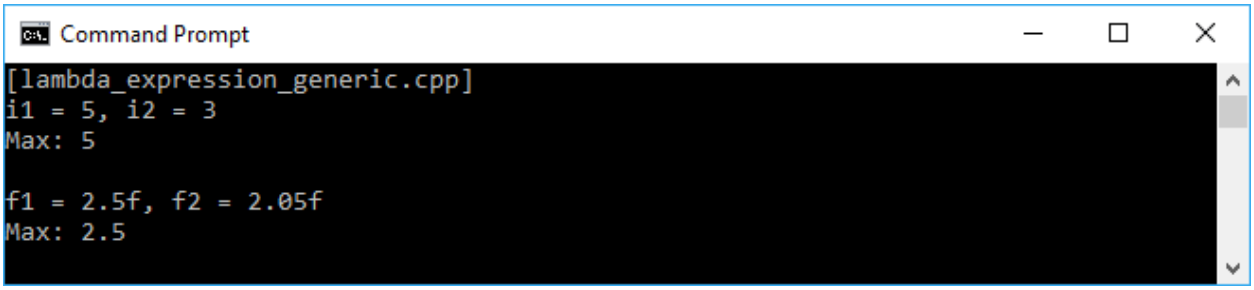

Command Prompt
[lambda_returning_value.cpp]
Original Data:
0 1 2 3 4 5 6 7 8 9
Squared Data:
0 1 4 9 16 25 36 49 64 81
Average Data:
0 0.5 2 4.5 8 12.5 18 24.5 32 40.5

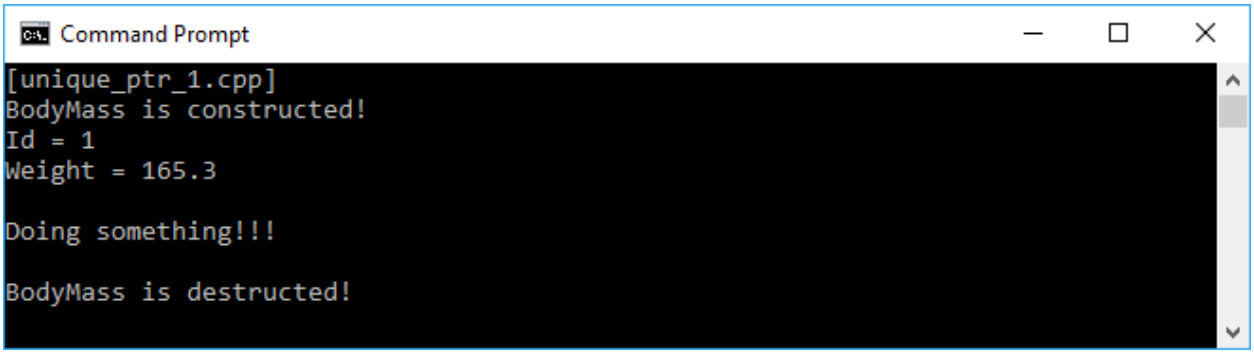

Command Prompt
[lambda_capturing_by_value.cpp]
Original Data:
0 1 2 3 4 5 6 7 8 9
Printing elements between 2 and 8 explicitly [a,b]:
2 3 4 5 6 7 8
printing elements between 3 and 7 implicitly[=]:
3 4 5 6 7

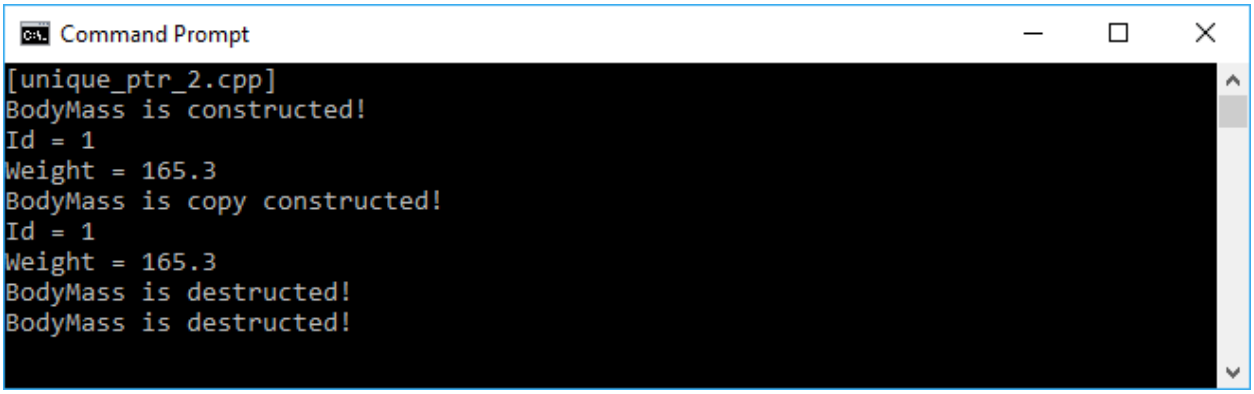

Command Prompt
[lambda_capturing_by_value_mutable.cpp]
Original Data:
0 1 2 3 4 5 6 7 8 9
Squared Data:
0 2 4 6 8 10 12 14 16 18
a = 1
b = 1

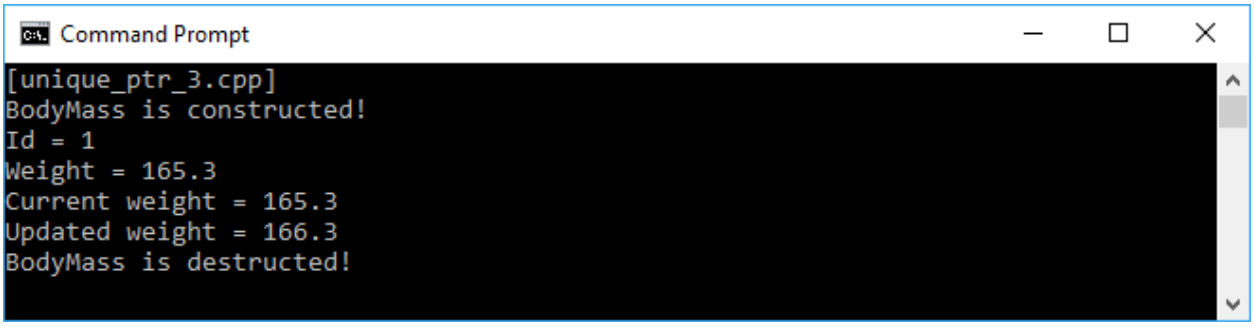

Command Prompt
[lambda_capturing_by_reference.cpp]
Original Data:
0 1 2 3 4 5 6 7 8 9
Squared Data:
0 2 4 6 8 10 12 14 16 18
a = 8
b = 9

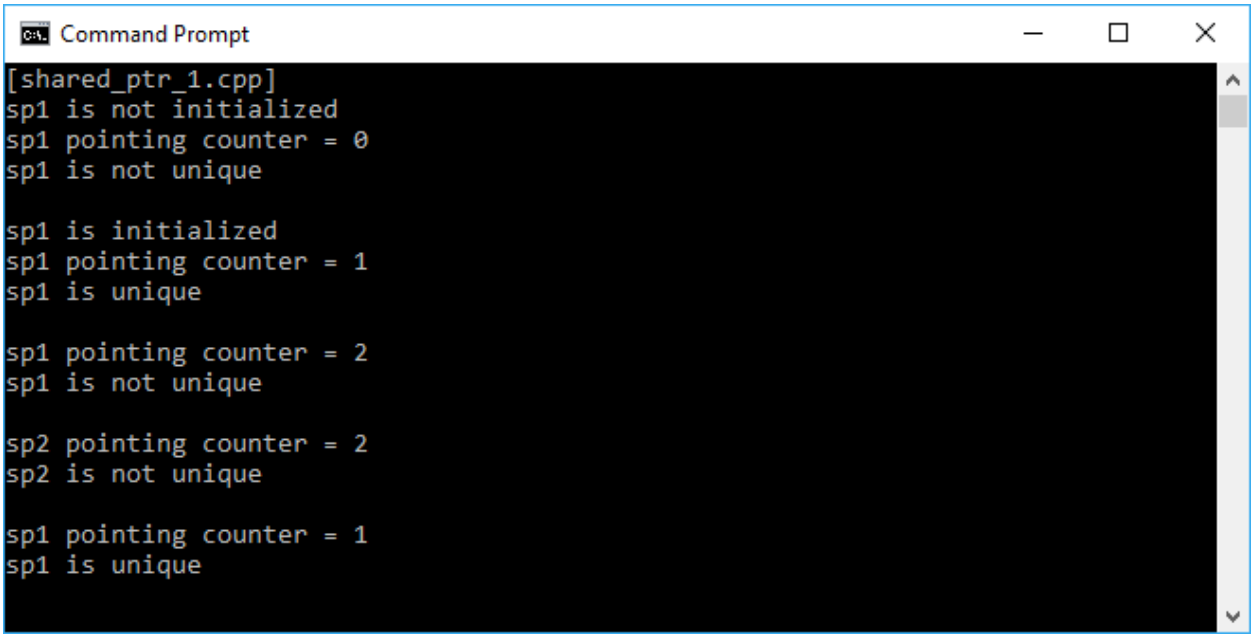

Command Prompt
[lambda_initialization_captures.cpp]
Initial a = 5
New a = 7

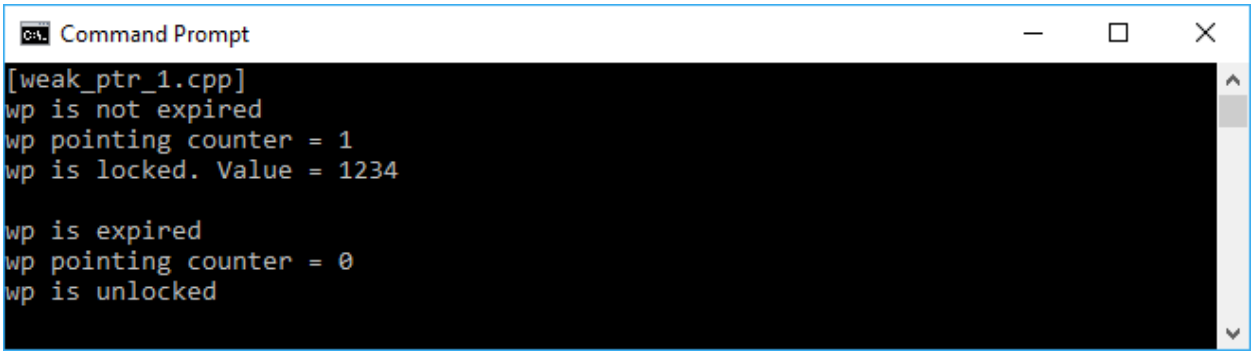

Command Prompt
[lambda_expression_generic.cpp]
i1 = 5, i2 = 3
Max: 5
f1 = 2.5f, f2 = 2.05f
Max: 2.5

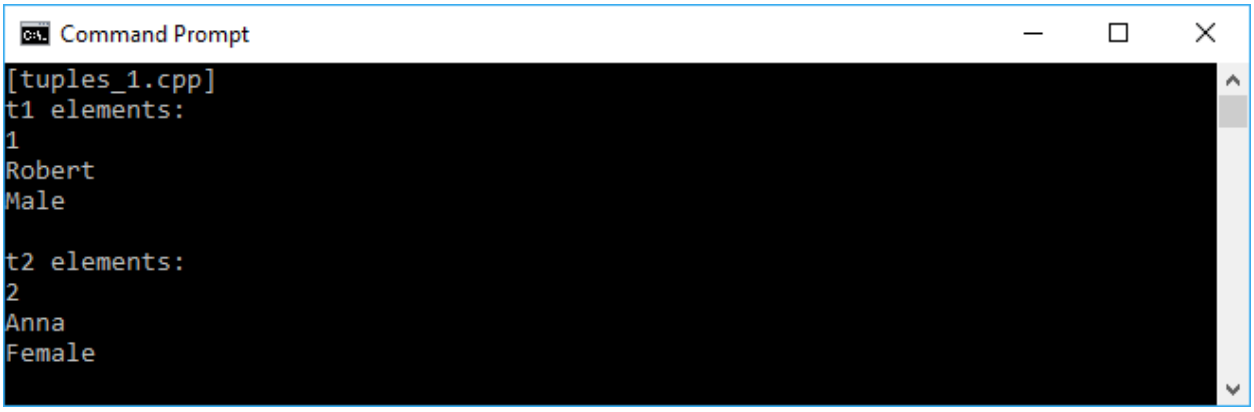

Command Prompt
[unique_ptr_1.cpp]
BodyMass is constructed!
Id = 1
Weight = 165.3
Doing something!!!
BodyMass is destructed!

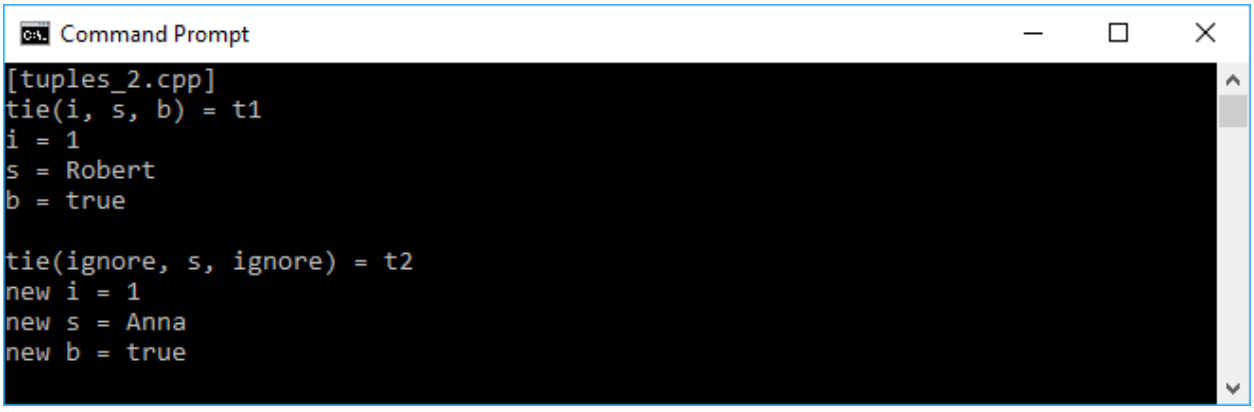

Command Prompt
[unique_ptr_2.cpp]
BodyMass is constructed!
Id = 1
Weight = 165.3
BodyMass is copy constructed!
Id = 1
Weight = 165.3
BodyMass is destructed!
BodyMass is destructed!

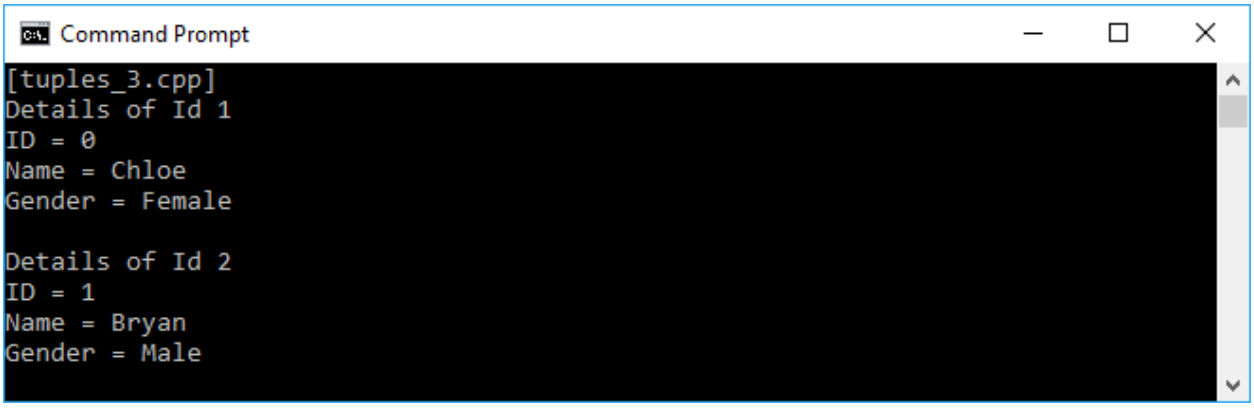

Command Prompt
[unique_ptr_3.cpp]
BodyMass is constructed!
Id = 1
Weight = 165.3
Current weight = 165.3
Updated weight = 166.3
BodyMass is destructed!


Command Prompt
[shared_ptr_1.cpp]
sp1 is not initialized
sp1 pointing counter = 0
sp1 is not unique
sp1 is initialized
sp1 pointing counter = 1
sp1 is unique
sp1 pointing counter = 2
sp1 is not unique
sp2 pointing counter = 2
sp2 is not unique
sp1 pointing counter = 1
sp1 is unique

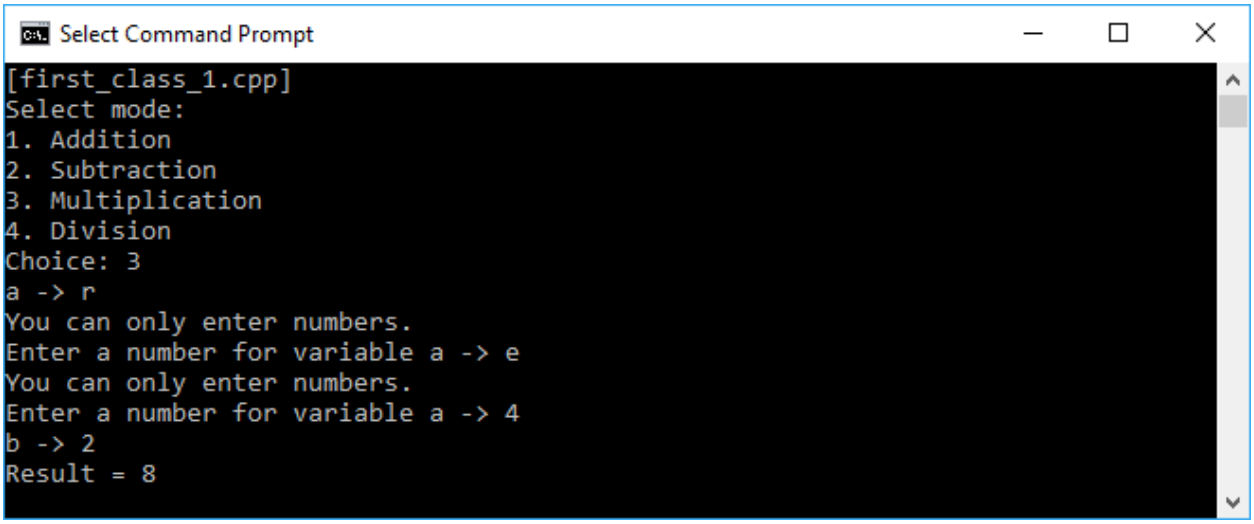

Command Prompt
[weak_ptr_1.cpp]
wp is not expired
wp pointing counter = 1
wp is locked. Value = 1234
wp is expired
wp pointing counter = 0
wp is unlocked

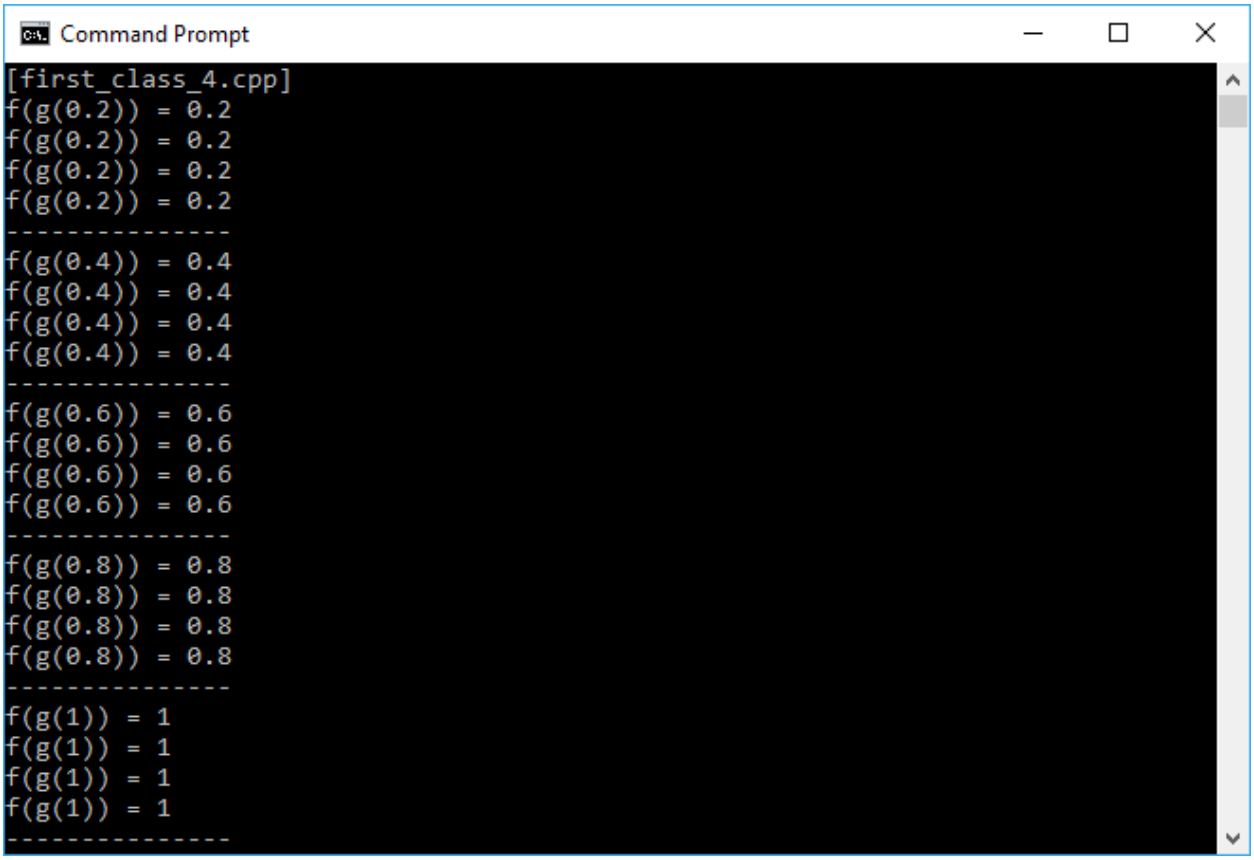

Command Prompt
[tuples_1.cpp]
t1 elements:
1
Robert
Male
t2 elements:
2
Anna
Female


Command Prompt
[tuples_2.cpp]
tie(i, s, b) = t1
i = 1
s = Robert
b = true
tie(ignore, s, ignore) = t2
new i = 1
new s = Anna
new b = true

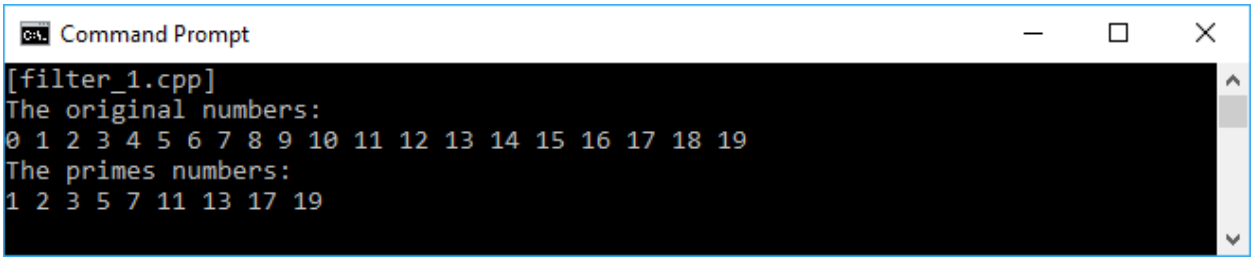

Command Prompt
[tuples_3.cpp]
Details of Id 1
ID = 0
Name = Chloe
Gender = Female
Details of Id 2
ID = 1
Name = Bryan
Gender = Male

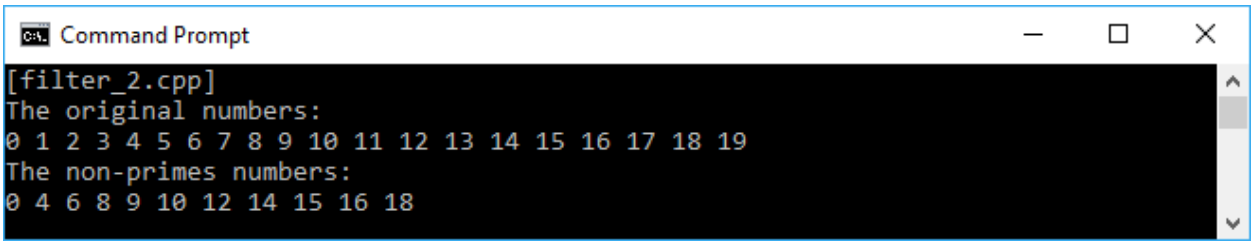
Chapter 2: Manipulating Functions in Functional Programming

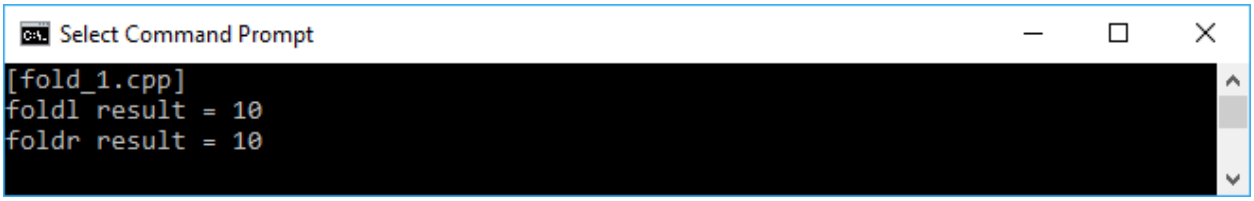


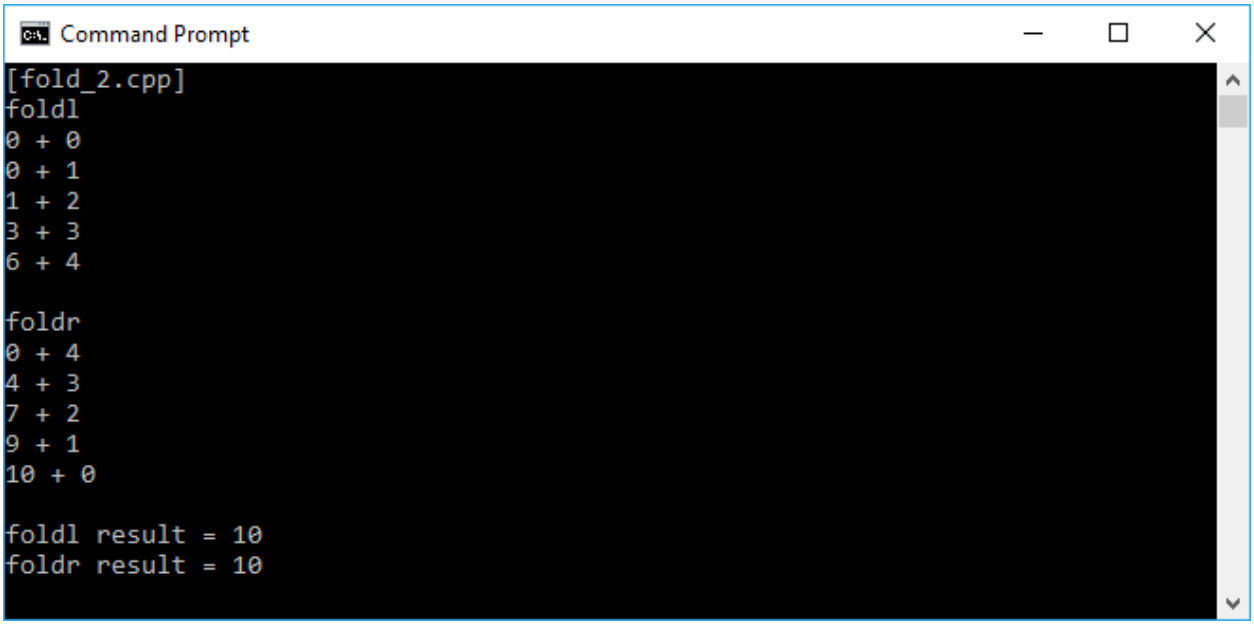

Command Prompt
[first_class_4.cpp]
f(g(0.2)) = 0.2
f(g(0.2)) = 0.2
f(g(0.2)) = 0.2
f(g(0.2)) = 0.2
---
f(g(0.4)) = 0.4
f(g(0.4)) = 0.4
f(g(0.4)) = 0.4
f(g(0.4)) = 0.4
---
f(g(0.6)) = 0.6
f(g(0.6)) = 0.6
f(g(0.6)) = 0.6
f(g(0.6)) = 0.6
---
f(g(0.8)) = 0.8
f(g(0.8)) = 0.8
f(g(0.8)) = 0.8
f(g(0.8)) = 0.8
---
f(g(1)) = 1
f(g(1)) = 1
f(g(1)) = 1
f(g(1)) = 1

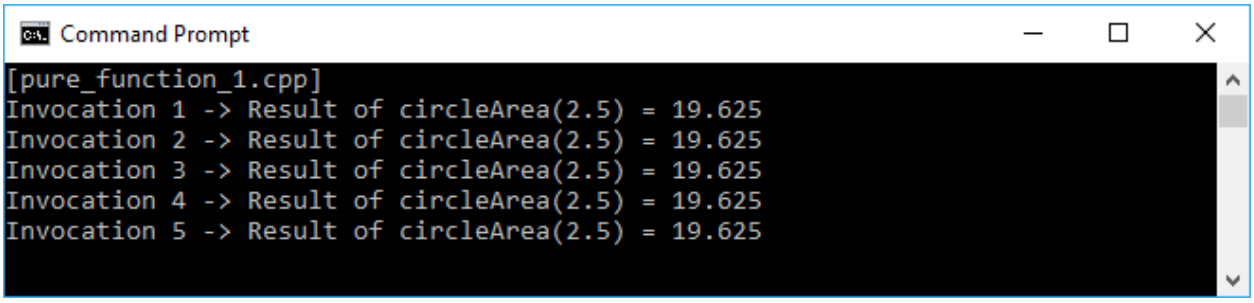

Command Prompt
[transform_1.cpp]
v1 contains: 0 1 2 3 4
v2 contains: 0 1 4 9 16

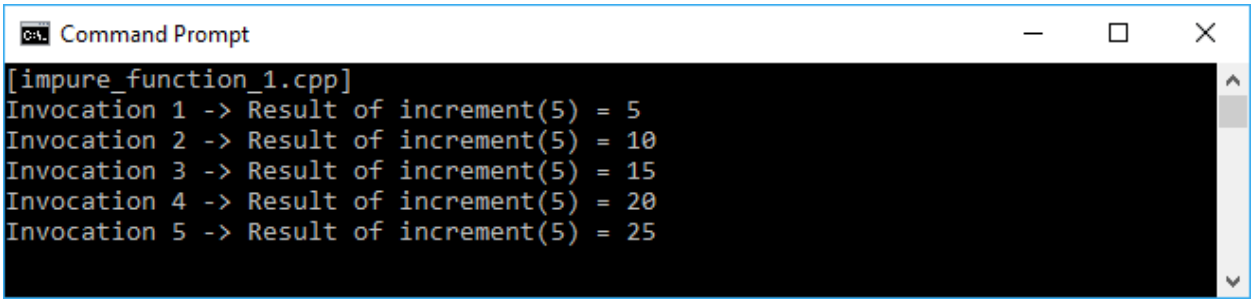

Command Prompt
[filter_1.cpp]
The original numbers:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
The primes numbers:
1 2 3 5 7 11 13 17 19

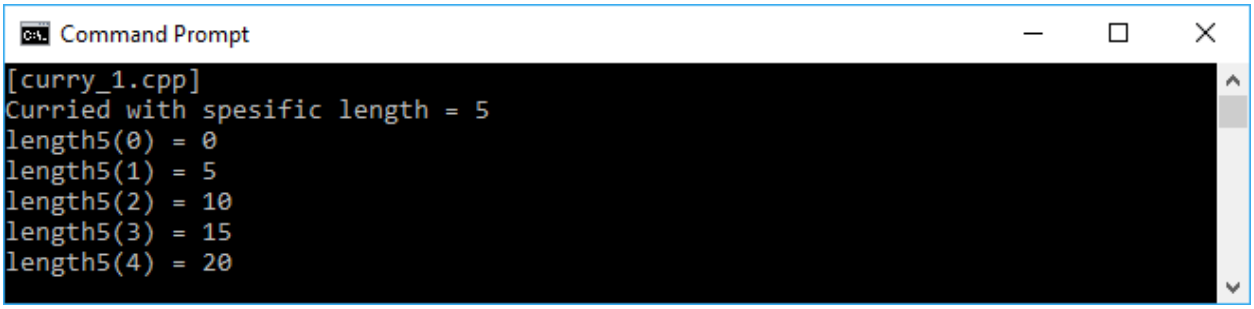

Command Prompt
[filter_2.cpp]
The original numbers:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
The non-primes numbers:
0 4 6 8 9 10 12 14 15 16 18

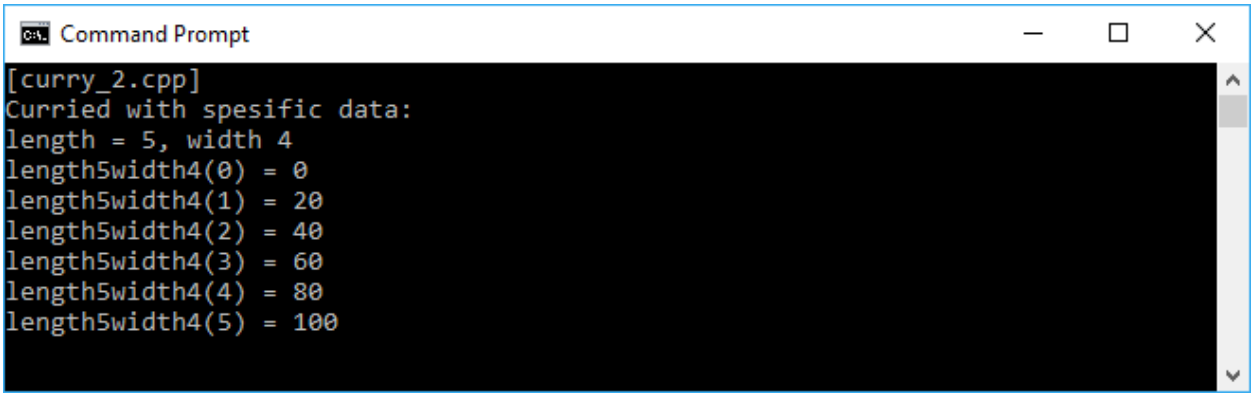

Select Command Prompt
[fold_1.cpp]
foldl result = 10
foldr result = 10


Command Prompt
[fold_2.cpp]
foldl
0 + 0
0 + 1
1 + 2
3 + 3
6 + 4
foldr
0 + 4
4 + 3
7 + 2
9 + 1
10 + 0
foldl result = 10
foldr result = 10

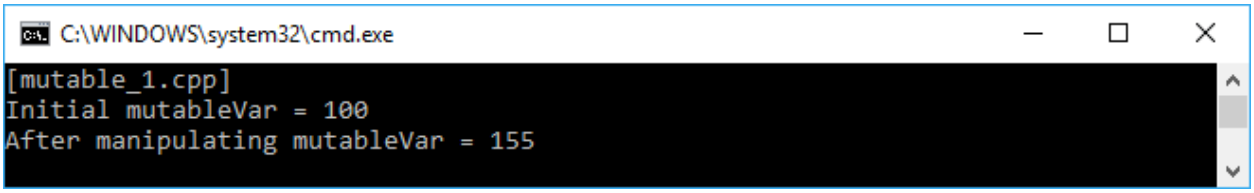

Command Prompt
[pure_function_1.cpp]
Invocation 1 -> Result of circleArea(2.5) = 19.625
Invocation 2 -> Result of circleArea(2.5) = 19.625
Invocation 3 -> Result of circleArea(2.5) = 19.625
Invocation 4 -> Result of circleArea(2.5) = 19.625
Invocation 5 -> Result of circleArea(2.5) = 19.625

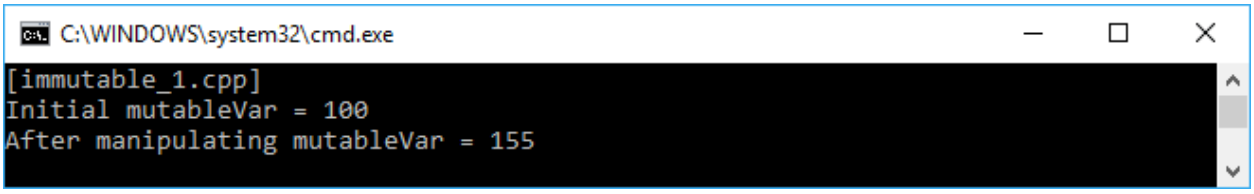

Command Prompt
[impure_function_1.cpp]
Invocation 1 -> Result of increment(5) = 5
Invocation 2 -> Result of increment(5) = 10
Invocation 3 -> Result of increment(5) = 15
Invocation 4 -> Result of increment(5) = 20
Invocation 5 -> Result of increment(5) = 25

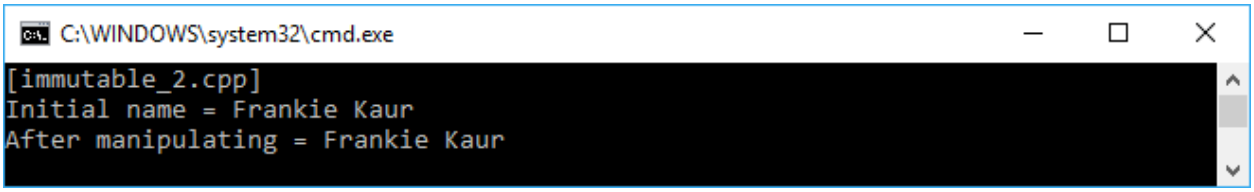

Command Prompt
[curry_1.cpp]
Curried with spesific length = 5
length5(0) = 0
length5(1) = 5
length5(2) = 10
length5(3) = 15
length5(4) = 20

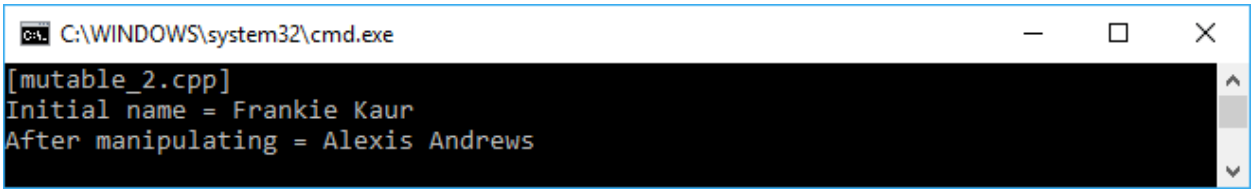

Command Prompt
[curry_2.cpp]
Curried with spesific data:
length = 5, width 4
length5width4(0) = 0
length5width4(1) = 20
length5width4(2) = 40
length5width4(3) = 60
length5width4(4) = 80
length5width4(5) = 100

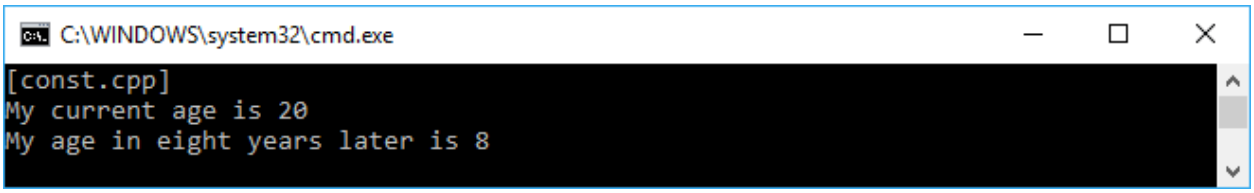
Chapter 3: Applying Immutable State to the Function

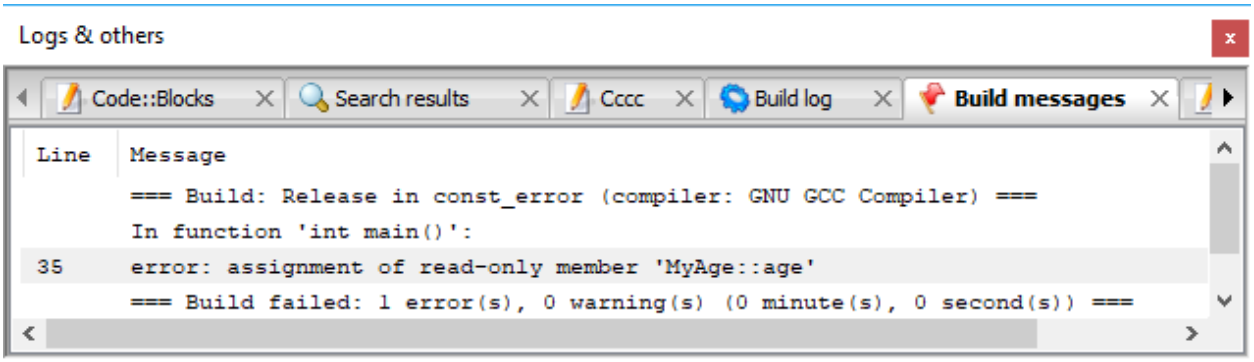


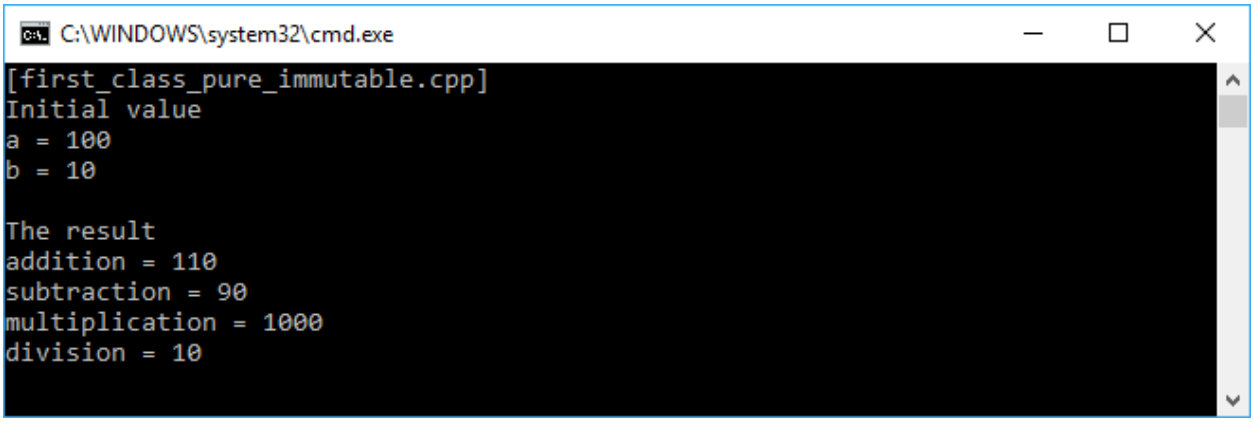


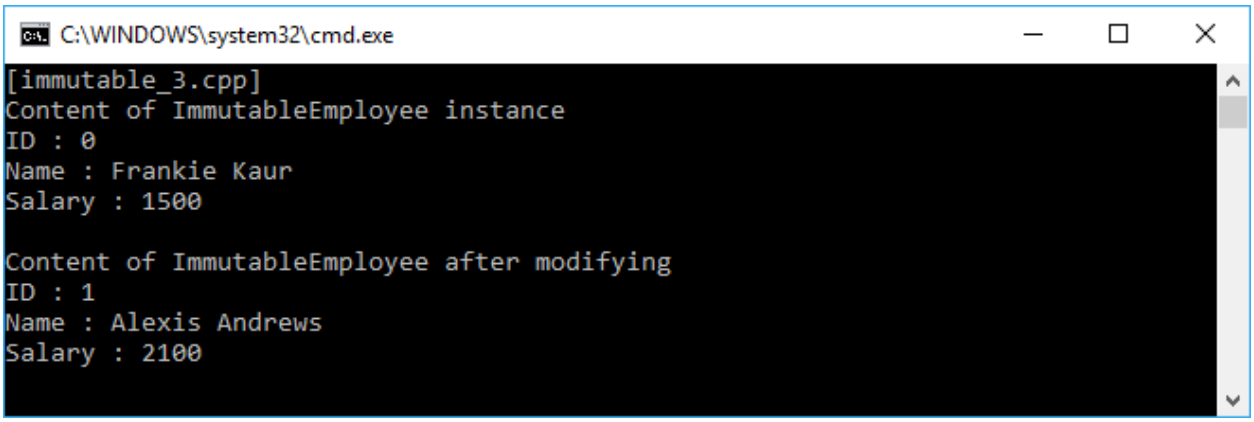


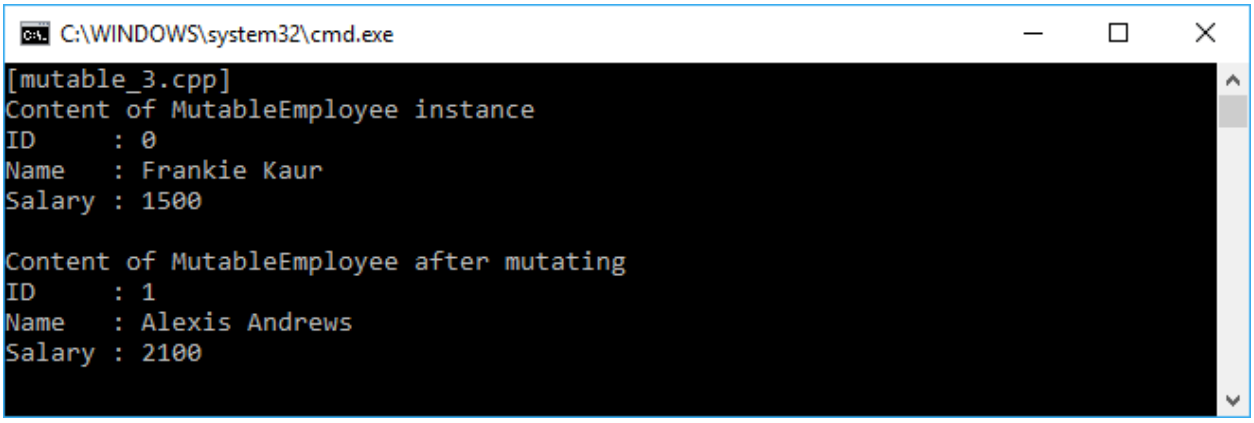

C:\WINDOWS\system32\cmd.exe
[mutable_2.cpp]
Initial name = Frankie Kaur
After manipulating = Alexis Andrews


C:\WINDOWS\system32\cmd.exe
[const.cpp]
My current age is 20
My age in eight years later is 8

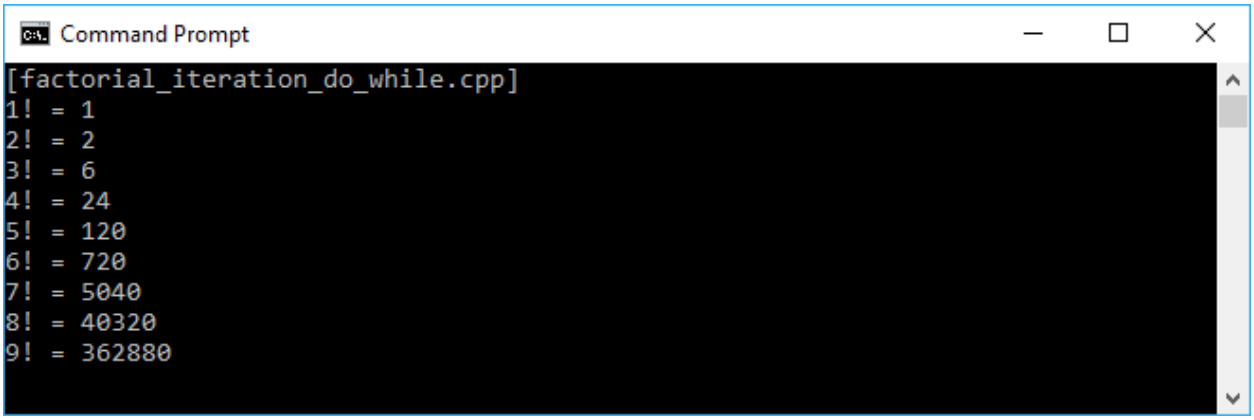

Logs & others
Code::Blocks
Search results
Cccc
Build log
Build messages
Line
Message
=== Build: Release in const_error (compiler: GNU GCC Compiler) ===
In function 'int main()':
35
error: assignment of read-only member 'MyAge::age'
=== Build failed: 1 error(s), 0 warning(s) (0 minute(s), 0 second(s)) ===

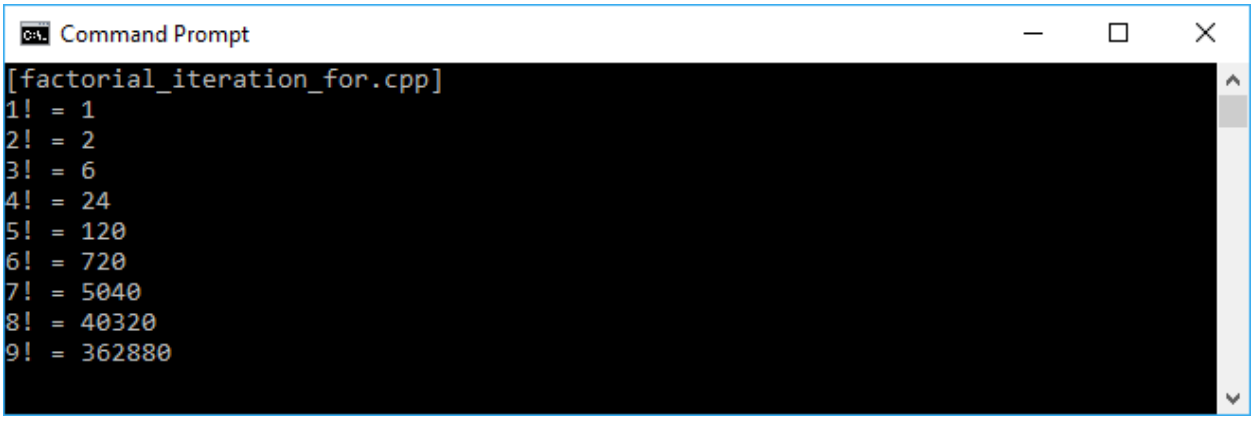

C:\WINDOWS\system32\cmd.exe
[first_class_pure_immutable.cpp]
Initial value
a = 100
b = 10
The result
addition = 110
subtraction = 90
multiplication = 1000
division = 10

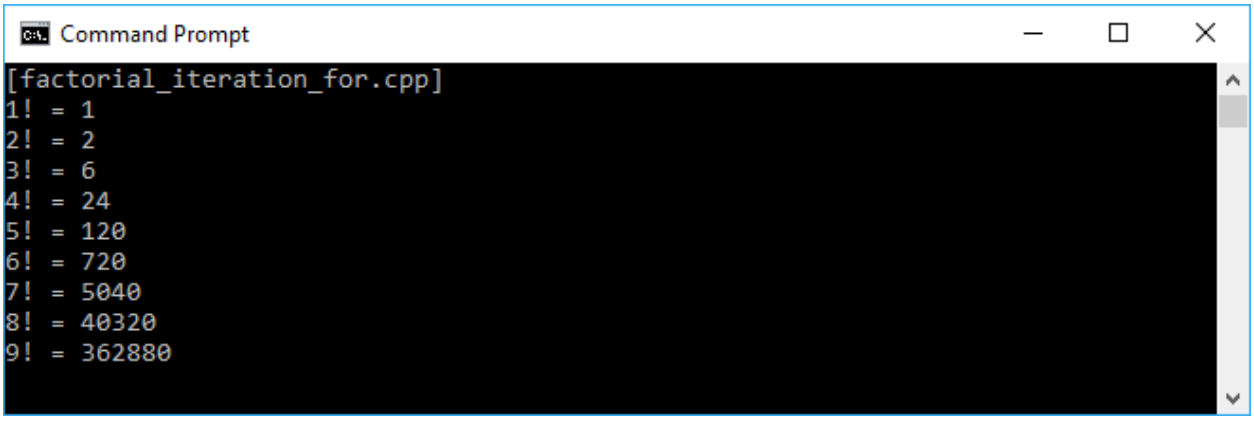

C:\WINDOWS\system32\cmd.exe
[immutable_3.cpp]
Content of ImmutableEmployee instance
ID : 0
Name : Frankie Kaur
Salary : 1500
Content of ImmutableEmployee after modifying
ID : 1
Name : Alexis Andrews
Salary : 2100


C:\WINDOWS\system32\cmd.exe
[mutable_3.cpp]
Content of MutableEmployee instance
ID   : 0
Name   : Frankie Kaur
Salary : 1500
Content of MutableEmployee after mutating
ID   : 1
Name   : Alexis Andrews
Salary : 2100

Chapter 4: Repeating Method Invocation Using Recursive Algorithm

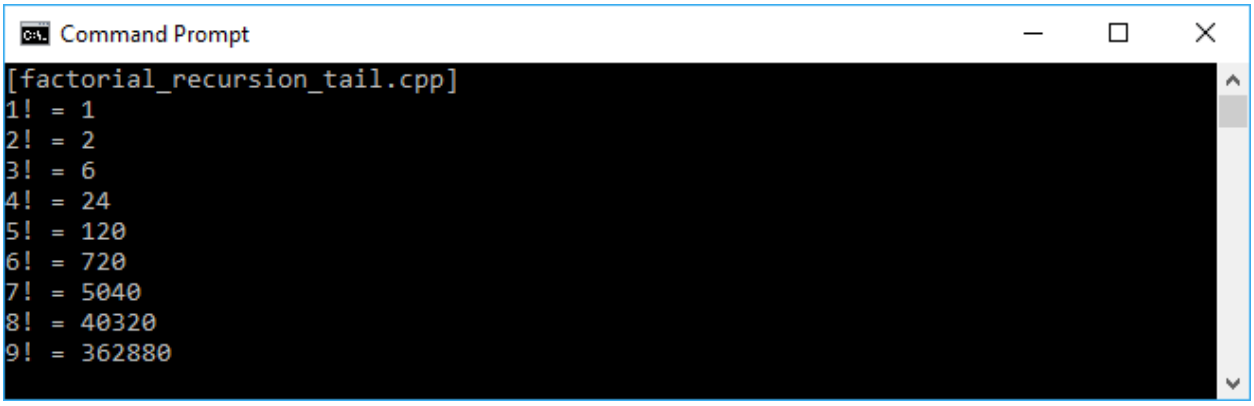


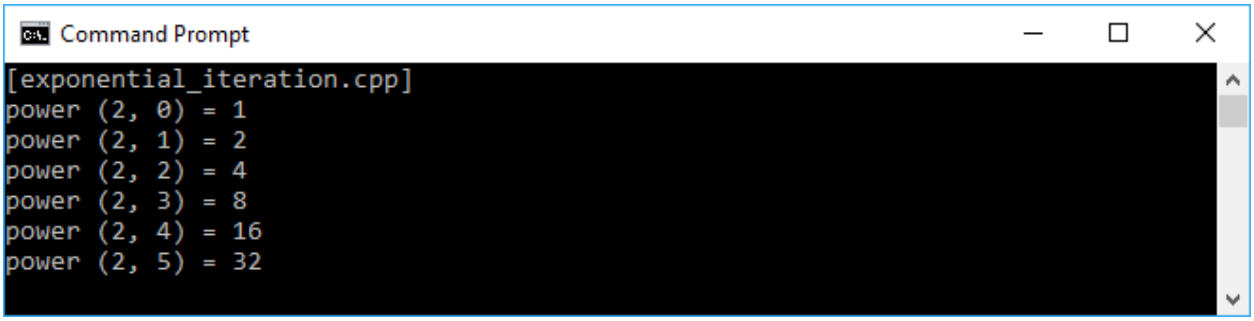


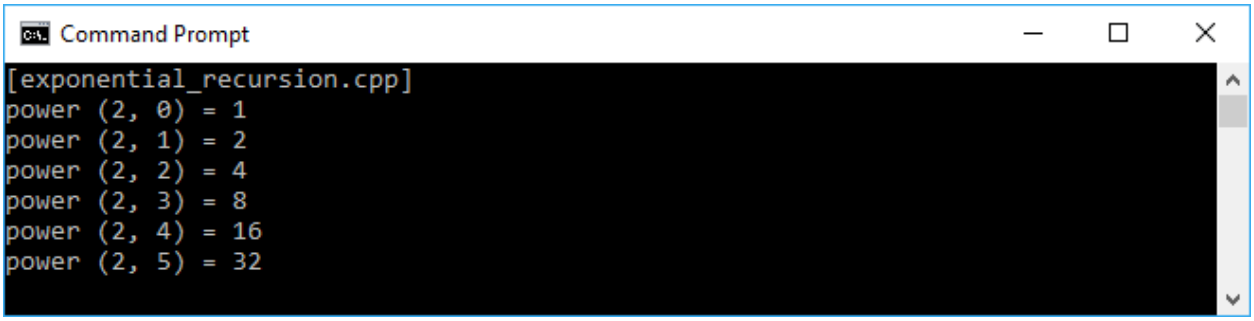

Command Prompt
[factorial_iteration_for.cpp]
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
7! = 5040
8! = 40320
9! = 362880

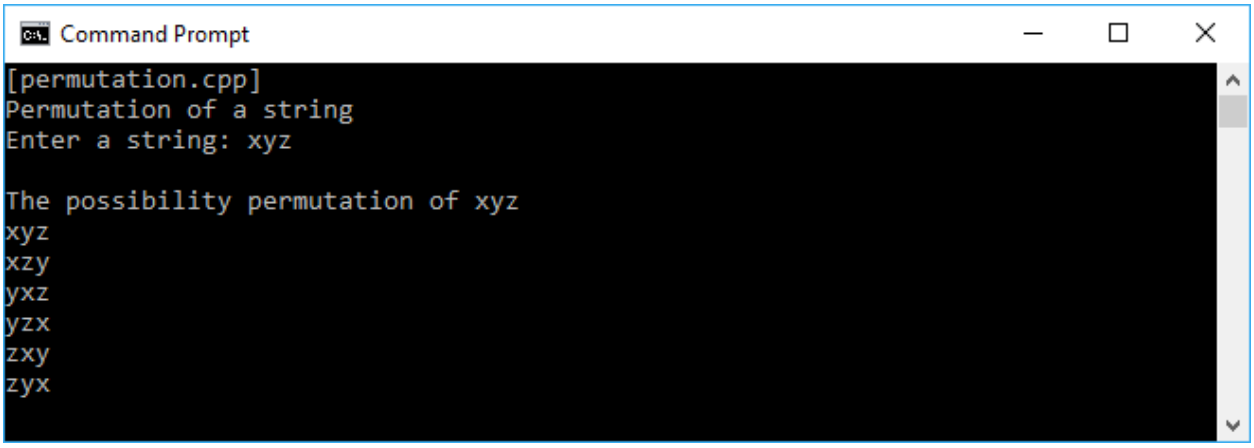

Command Prompt
[fibonacci_iteration.cpp]
0 1 1 2 3 5 8 13 21 34

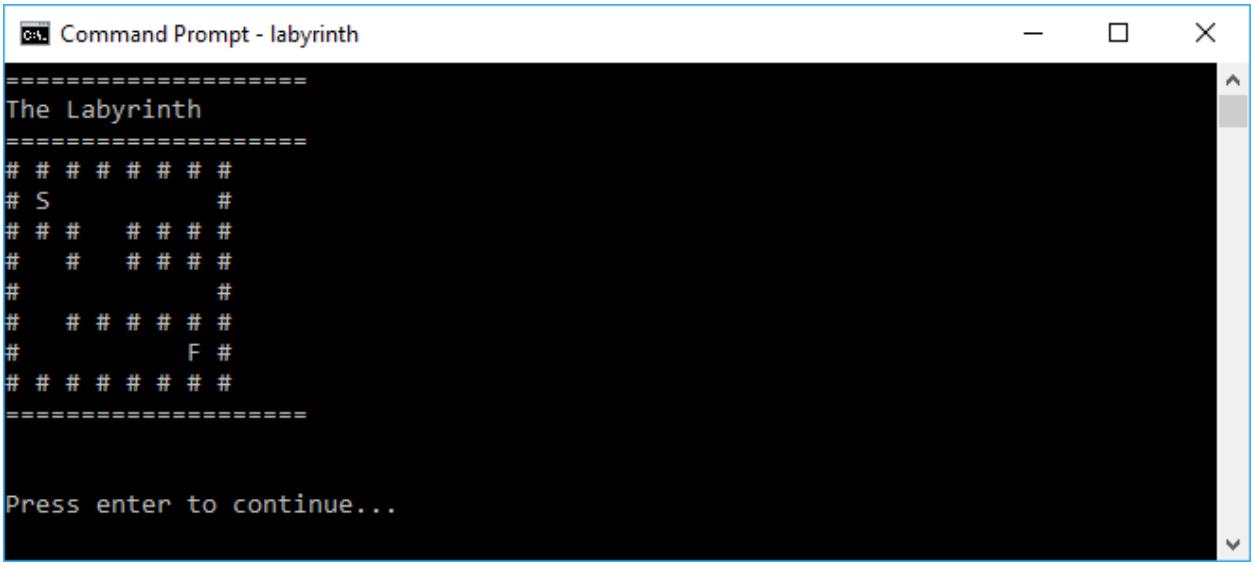

Command Prompt
[fibonacci_recursion.cpp]
0 1 1 2 3 5 8 13 21 34

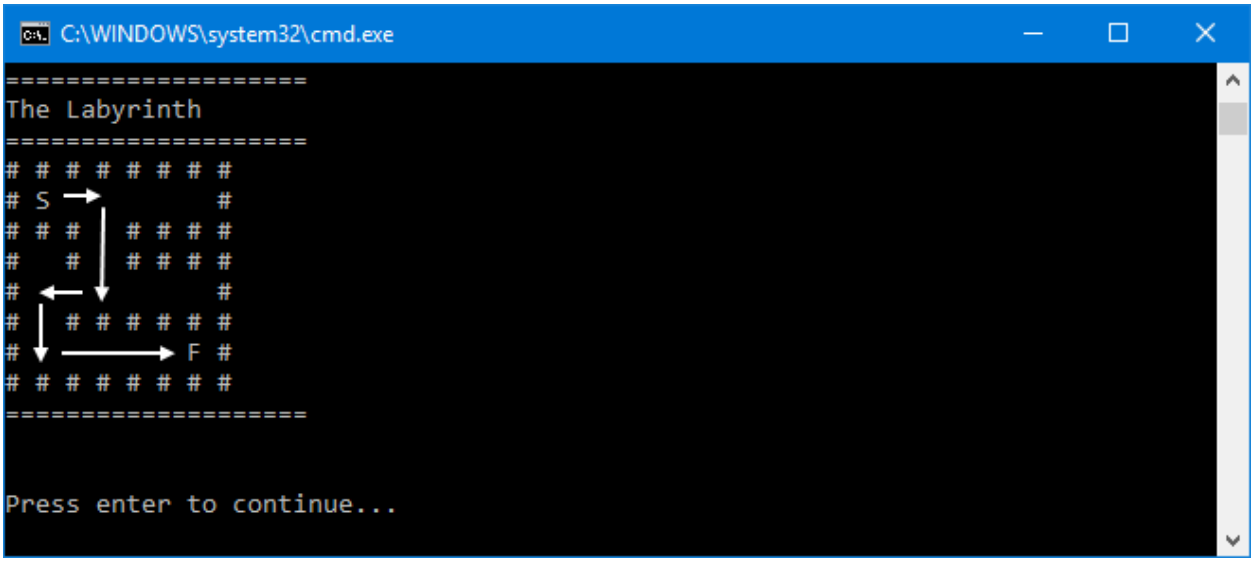

Command Prompt
[factorial_recursion_tail.cpp]
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
7! = 5040
8! = 40320
9! = 362880

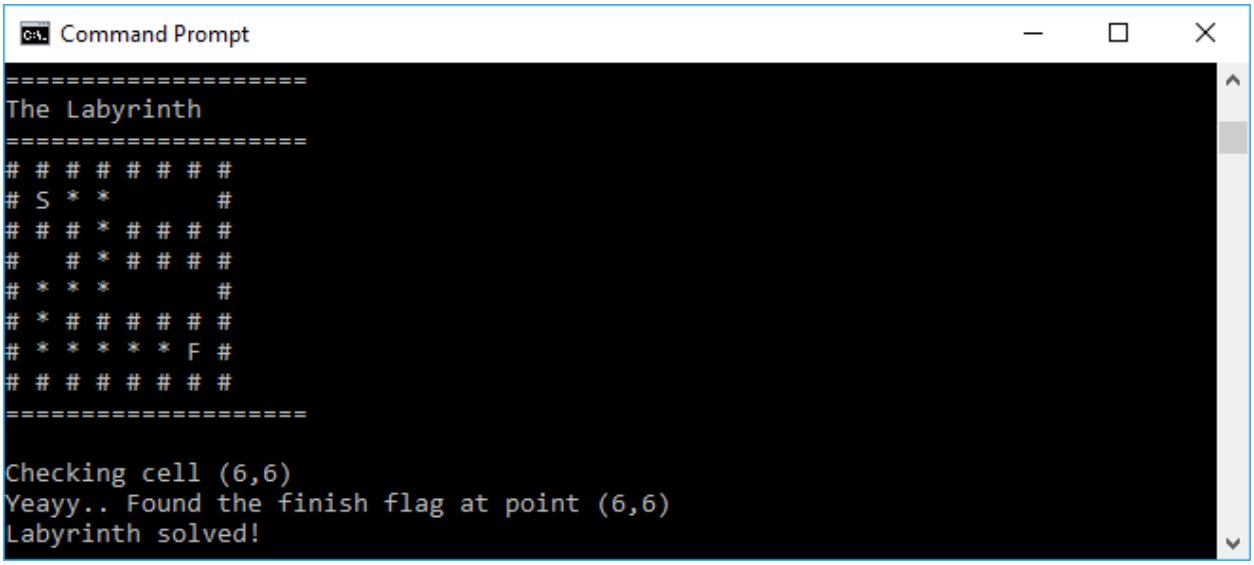

Command Prompt
[exponential_iteration.cpp]
power (2, 0) = 1
power (2, 1) = 2
power (2, 2) = 4
power (2, 3) = 8
power (2, 4) = 16
power (2, 5) = 32

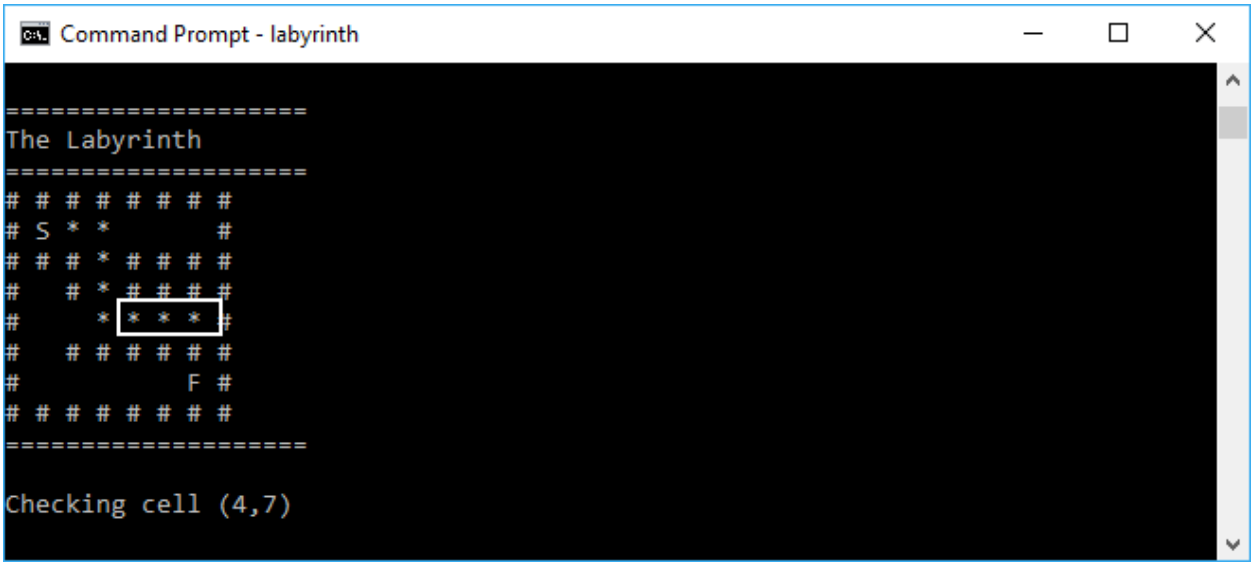

Command Prompt
[exponential_recursion.cpp]
power (2, 0) = 1
power (2, 1) = 2
power (2, 2) = 4
power (2, 3) = 8
power (2, 4) = 16
power (2, 5) = 32

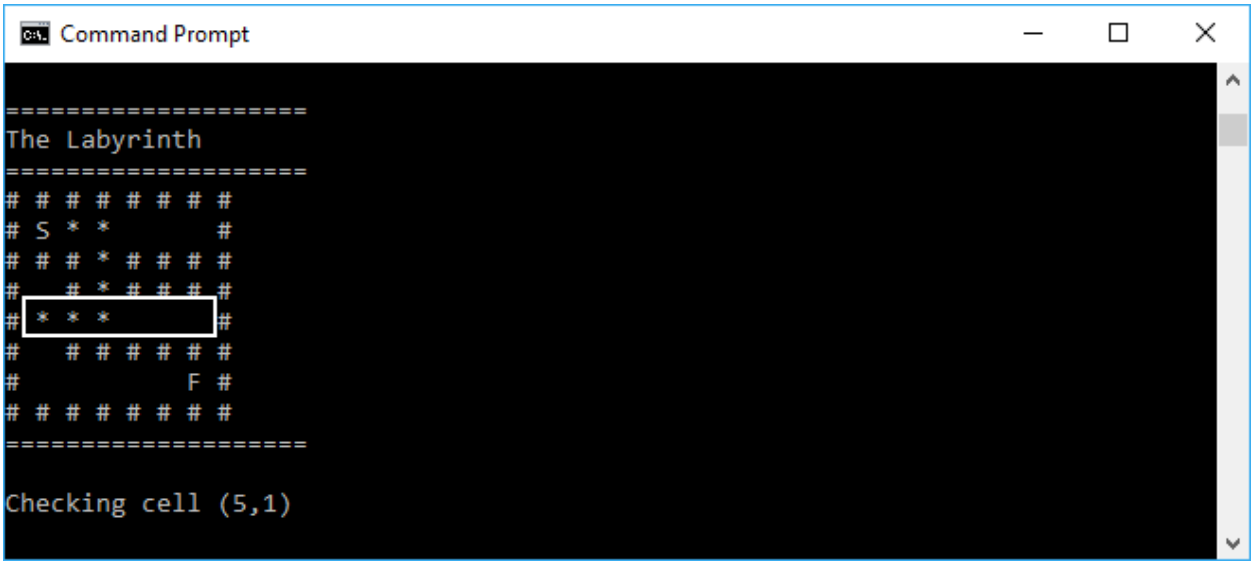

Command Prompt
[permutation.cpp]
Permutation of a string
Enter a string: xyz
The possibility permutation of xyz
xyz
xzy
yxz
yzx
zxy
zyx


Command Prompt - labyrinth
====================
The Labyrinth
====================
# # # # # # # #
S
# # #   # # # #
#   #   # # # #
#   #
#   # # # # # #
F
# # # # # # # #
====================
Press enter to continue...

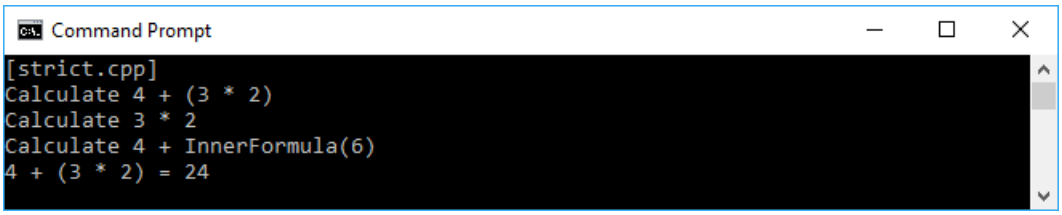

C:\WINDOWS\system32\cmd.exe
==========
The Labyrinth
==========
# # # # # # # #
S → #
| # # #
| # # #
← #
| # # # # #
↓ → F
# # # # # # # #
==========
Press enter to continue...

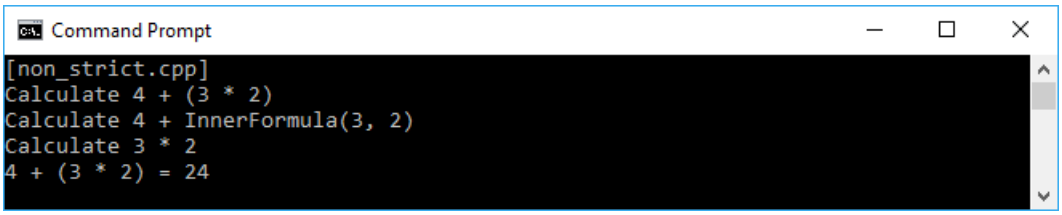

Command Prompt
==========
The Labyrinth
==========
# # # # # # # #
S * *
* # # #
* # # #
* * *
* # # # # #
* * * * * F
# # # # # # # #
==========
Checking cell (6,6)
Yeayy.. Found the finish flag at point (6,6)
Labyrinth solved!

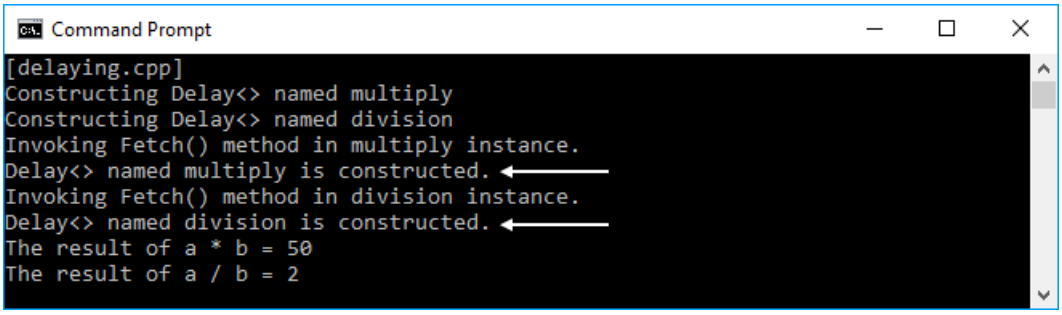

Command Prompt - labyrinth
====================
The Labyrinth
====================
# # # # # # # #
S * *
* # # #
* # # #
* * * *
#   # # # # # #
F
# # # # # # # #
====================
Checking cell (4,7)

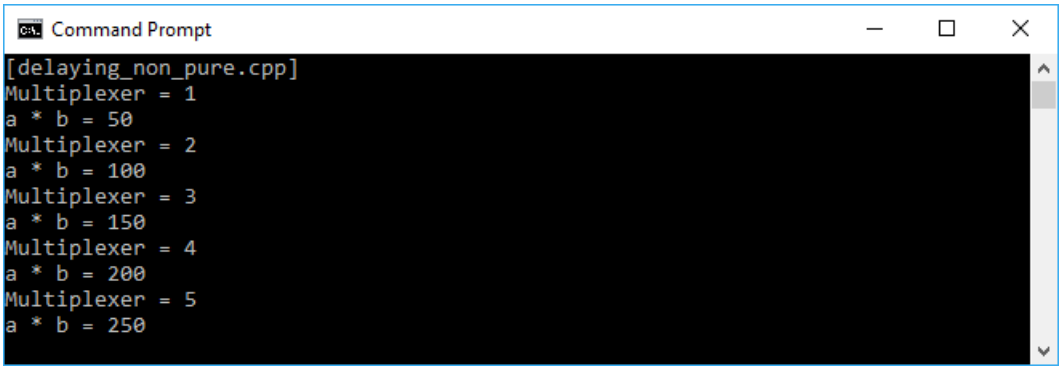

Command Prompt
====================
The Labyrinth
====================
# # # # # # # #
S * *
* # # #
* # # #
* * *
#   # # # # # #
F
# # # # # # # #
====================
Checking cell (5,1)

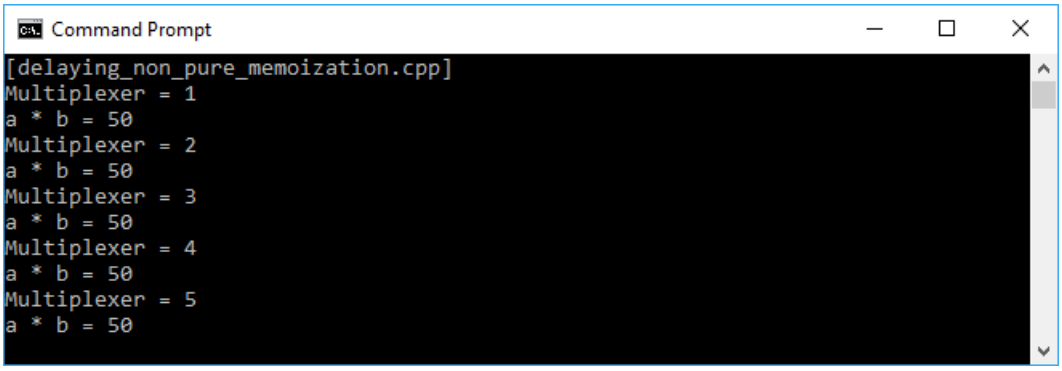
Chapter 5: Procrastinating the execution process using Lazy Evaluation

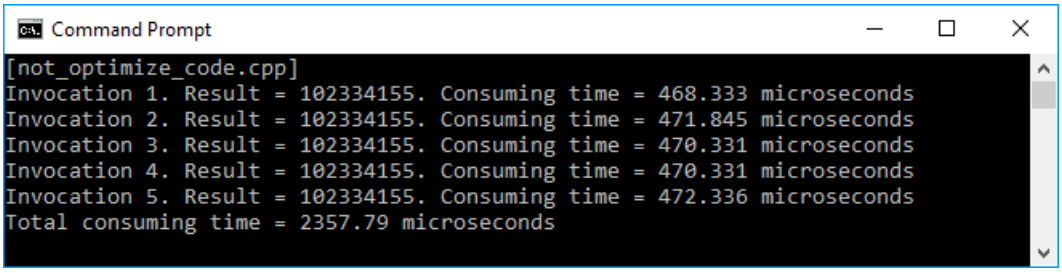


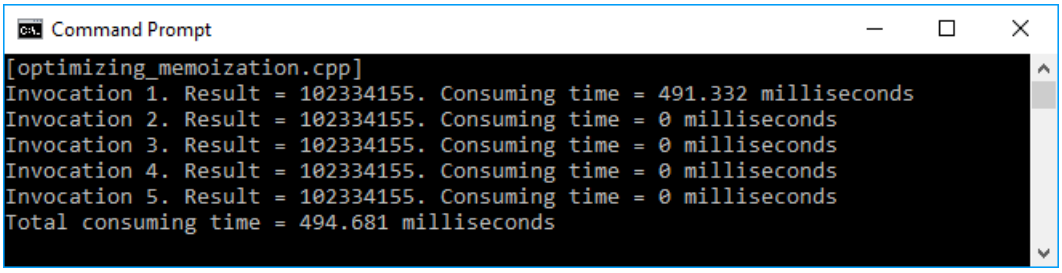


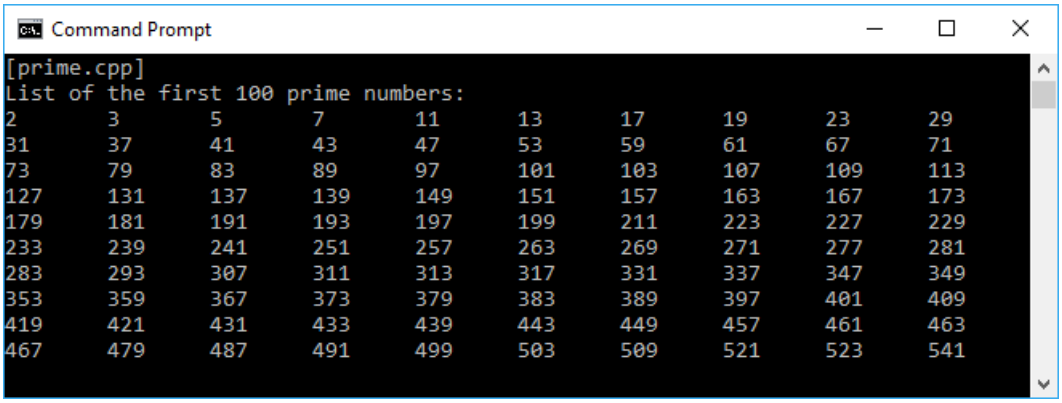


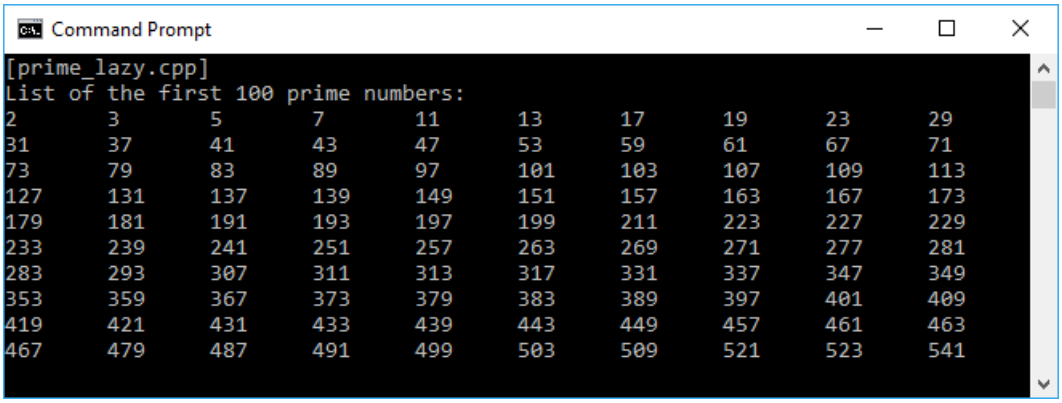

Command Prompt
[delaying_non_pure.cpp]
Multiplexer = 1
a * b = 50
Multiplexer = 2
a * b = 100
Multiplexer = 3
a * b = 150
Multiplexer = 4
a * b = 200
Multiplexer = 5
a * b = 250


Command Prompt
[delaying_non_pure_memoization.cpp]
Multiplexer = 1
a * b = 50
Multiplexer = 2
a * b = 50
Multiplexer = 3
a * b = 50
Multiplexer = 4
a * b = 50
Multiplexer = 5
a * b = 50

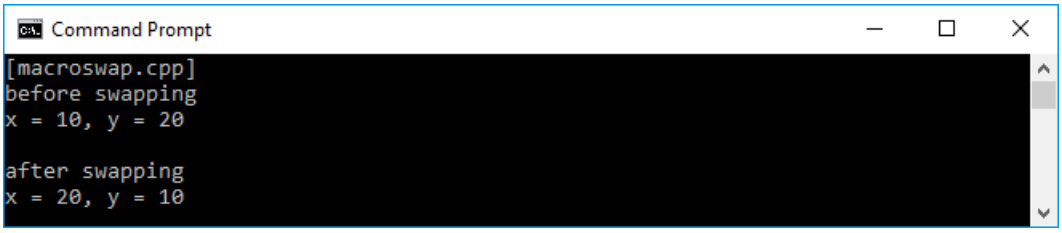

Command Prompt
[not_optimize_code.cpp]
Invocation 1. Result = 102334155. Consuming time = 468.333 microseconds
Invocation 2. Result = 102334155. Consuming time = 471.845 microseconds
Invocation 3. Result = 102334155. Consuming time = 470.331 microseconds
Invocation 4. Result = 102334155. Consuming time = 470.331 microseconds
Invocation 5. Result = 102334155. Consuming time = 472.336 microseconds
Total consuming time = 2357.79 microseconds


Command Prompt
[optimizing_memoization.cpp]
Invocation 1. Result = 102334155. Consuming time = 491.332 milliseconds
Invocation 2. Result = 102334155. Consuming time = 0 milliseconds
Invocation 3. Result = 102334155. Consuming time = 0 milliseconds
Invocation 4. Result = 102334155. Consuming time = 0 milliseconds
Invocation 5. Result = 102334155. Consuming time = 0 milliseconds
Total consuming time = 494.681 milliseconds

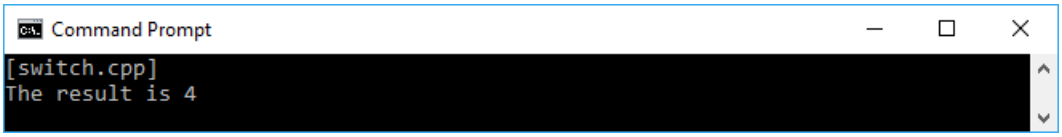

Command Prompt
[prime.cpp]
List of the first 100 prime numbers:
2 3 5 7 11 13 17 19 23 29
31 37 41 43 47 53 59 61 67 71
73 79 83 89 97 101 103 107 109 113
127 131 137 139 149 151 157 163 167 173
179 181 191 193 197 199 211 223 227 229
233 239 241 251 257 263 269 271 277 281
283 293 307 311 313 317 331 337 347 349
353 359 367 373 379 383 389 397 401 409
419 421 431 433 439 443 449 457 461 463
467 479 487 491 499 503 509 521 523 541

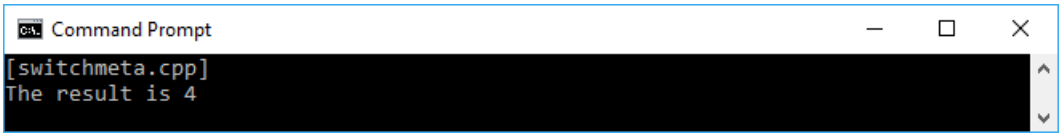

Command Prompt
[prime_lazy.cpp]
List of the first 100 prime numbers:
2 3 5 7 11 13 17 19 23 29
31 37 41 43 47 53 59 61 67 71
73 79 83 89 97 101 103 107 109 113
127 131 137 139 149 151 157 163 167 173
179 181 191 193 197 199 211 223 227 229
233 239 241 251 257 263 269 271 277 281
283 293 307 311 313 317 331 337 347 349
353 359 367 373 379 383 389 397 401 409
419 421 431 433 439 443 449 457 461 463
467 479 487 491 499 503 509 521 523 541

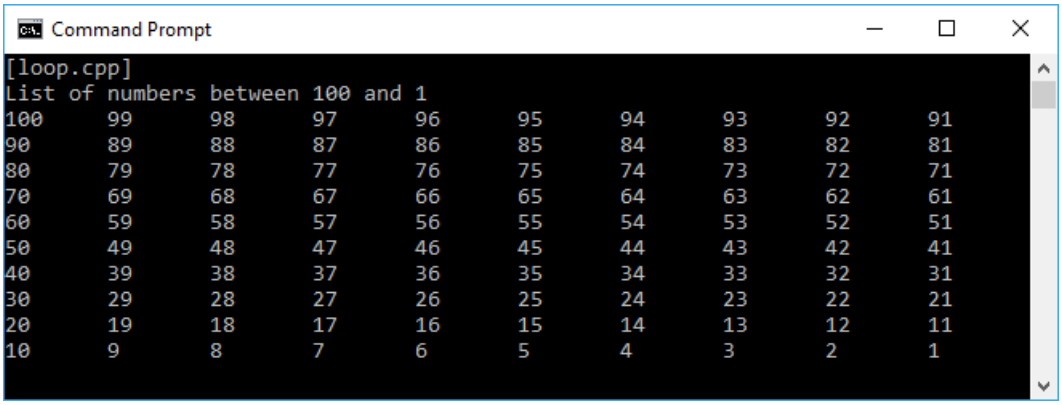
Chapter 6: Optimizing code with Metaprogramming

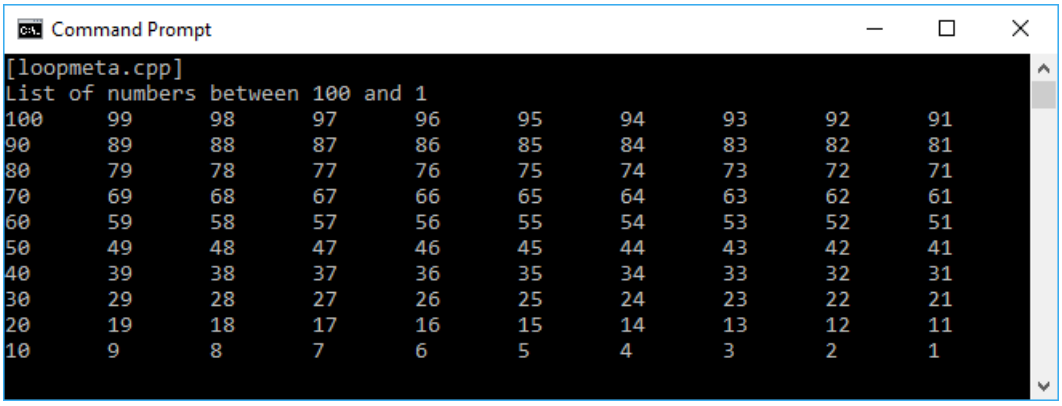


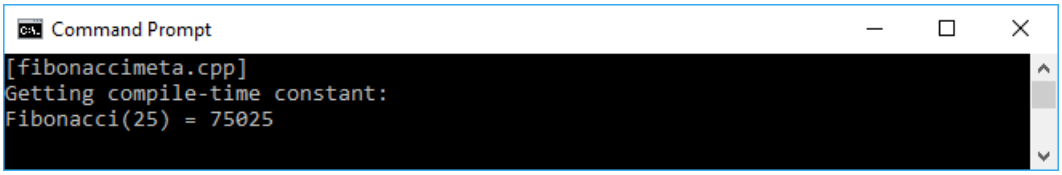

Command Prompt
[conditionmeta.cpp]
True Statement is run.

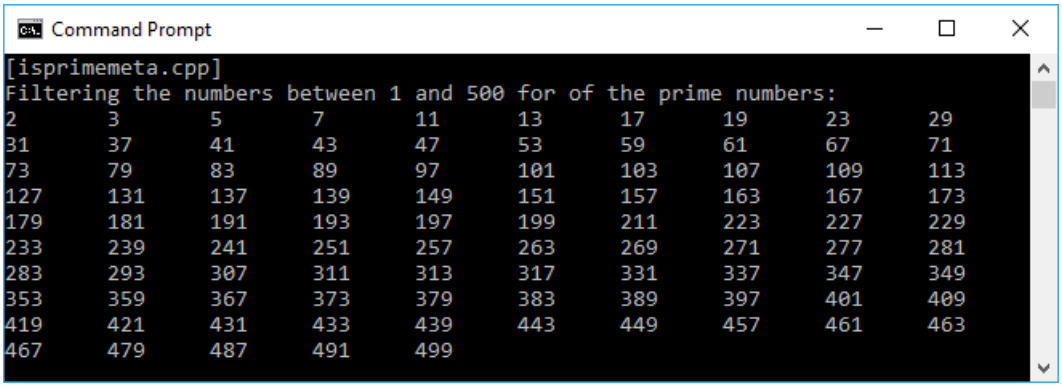

Command Prompt
[switch.cpp]
The result is 4


Command Prompt
[switchmeta.cpp]
The result is 4

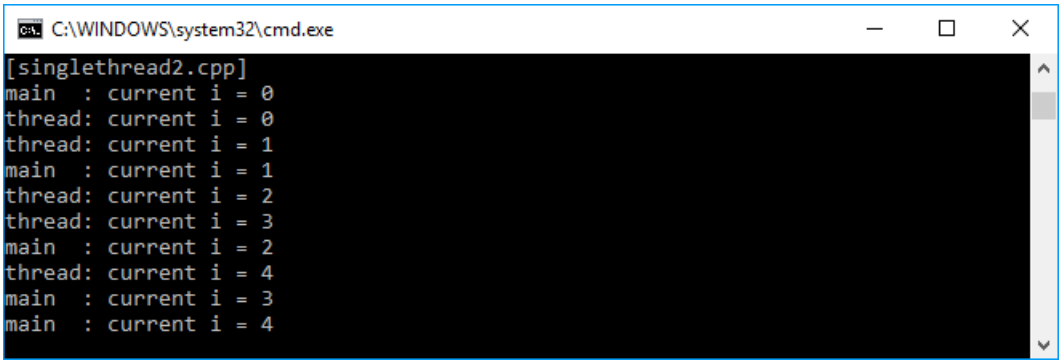

Command Prompt
[loop.cpp]
List of numbers between 100 and 1
100 99 98 97 96 95 94 93 92 91
90 89 88 87 86 85 84 83 82 81
80 79 78 77 76 75 74 73 72 71
70 69 68 67 66 65 64 63 62 61
60 59 58 57 56 55 54 53 52 51
50 49 48 47 46 45 44 43 42 41
40 39 38 37 36 35 34 33 32 31
30 29 28 27 26 25 24 23 22 21
20 19 18 17 16 15 14 13 12 11
10 9 8 7 6 5 4 3 2 1

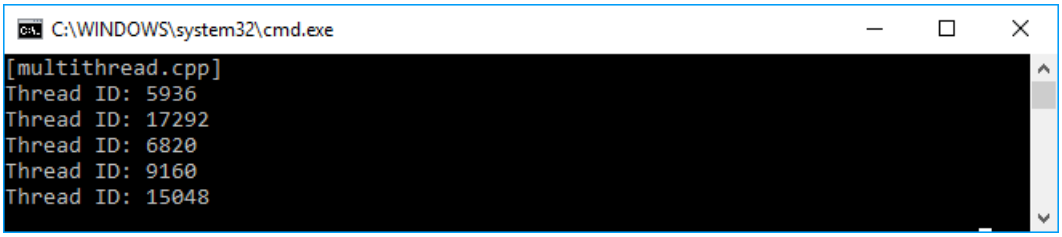

Command Prompt
[loopmeta.cpp]
List of numbers between 100 and 1
100 99 98 97 96 95 94 93 92 91
90 89 88 87 86 85 84 83 82 81
80 79 78 77 76 75 74 73 72 71
70 69 68 67 66 65 64 63 62 61
60 59 58 57 56 55 54 53 52 51
50 49 48 47 46 45 44 43 42 41
40 39 38 37 36 35 34 33 32 31
30 29 28 27 26 25 24 23 22 21
20 19 18 17 16 15 14 13 12 11
10 9 8 7 6 5 4 3 2 1

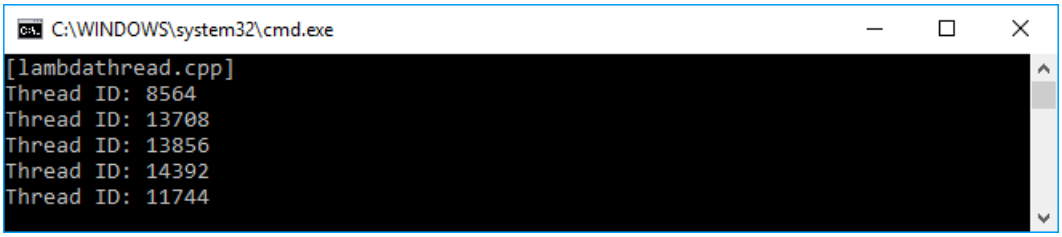

Command Prompt
[fibonaccimeta.cpp]
Getting compile-time constant:
Fibonacci(25) = 75025

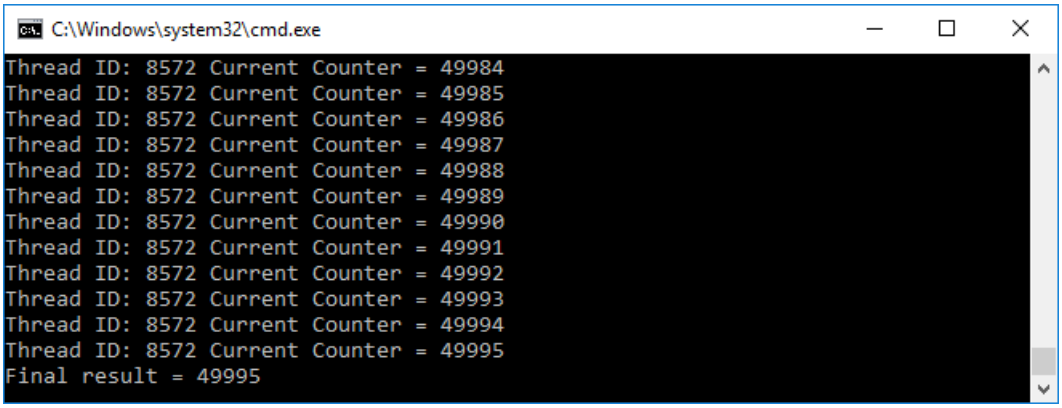

Command Prompt
[isprimemeta.cpp]
Filtering the numbers between 1 and 500 for of the prime numbers:
2 3 5 7 11 13 17 19 23 29
31 37 41 43 47 53 59 61 67 71
73 79 83 89 97 101 103 107 109 113
127 131 137 139 149 151 157 163 167 173
179 181 191 193 197 199 211 223 227 229
233 239 241 251 257 263 269 271 277 281
283 293 307 311 313 317 331 337 347 349
353 359 367 373 379 383 389 397 401 409
419 421 431 433 439 443 449 457 461 463
467 479 487 491 499

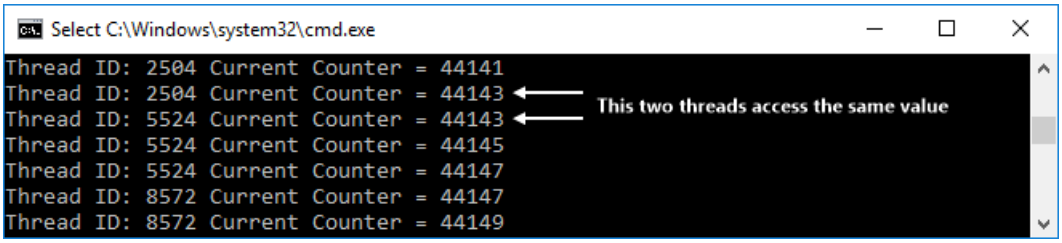
Chapter 7: Running parallel execution using Concurrency

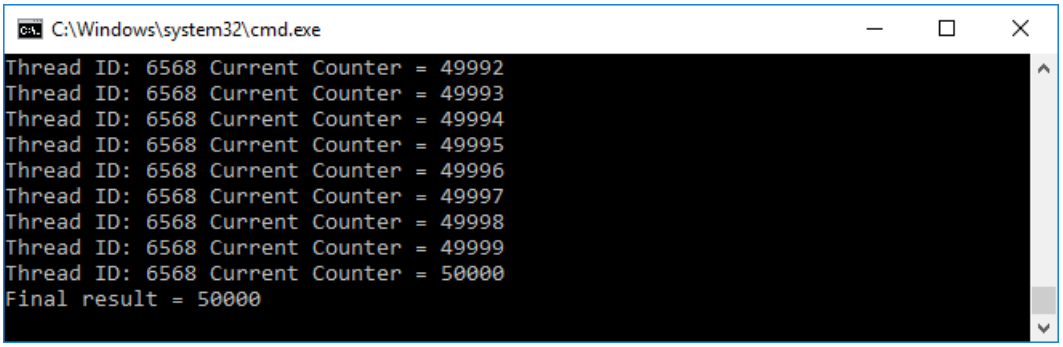


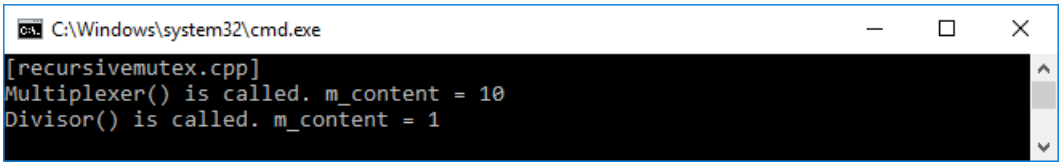



C:\WINDOWS\system32\cmd.exe
[lambdathread.cpp]
Thread ID: 8564
Thread ID: 13708
Thread ID: 13856
Thread ID: 14392
Thread ID: 11744

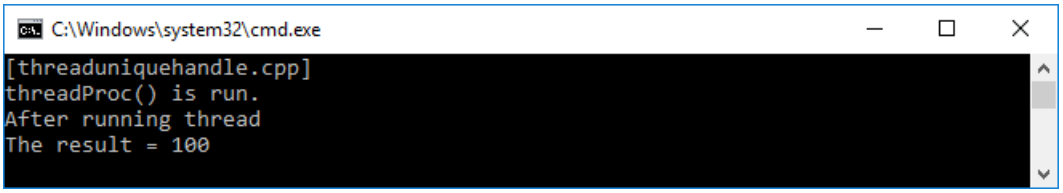

C:\Windows\system32\cmd.exe
Thread ID: 8572 Current Counter = 49984
Thread ID: 8572 Current Counter = 49985
Thread ID: 8572 Current Counter = 49986
Thread ID: 8572 Current Counter = 49987
Thread ID: 8572 Current Counter = 49988
Thread ID: 8572 Current Counter = 49989
Thread ID: 8572 Current Counter = 49990
Thread ID: 8572 Current Counter = 49991
Thread ID: 8572 Current Counter = 49992
Thread ID: 8572 Current Counter = 49993
Thread ID: 8572 Current Counter = 49994
Thread ID: 8572 Current Counter = 49995
Final result = 49995

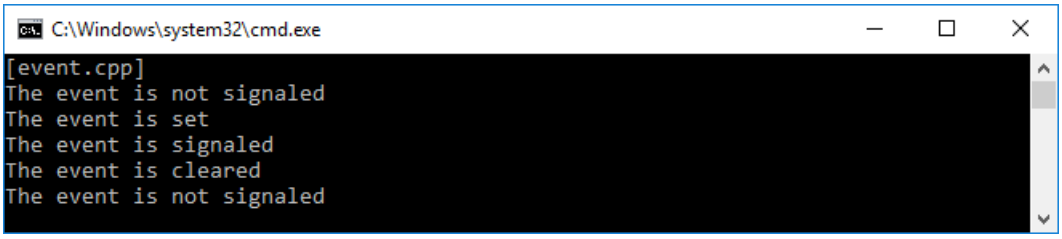

Select C:\Windows\system32\cmd.exe
Thread ID: 2504 Current Counter = 44141
Thread ID: 2504 Current Counter = 44143
Thread ID: 5524 Current Counter = 44143
This two threads access the same value
Thread ID: 5524 Current Counter = 44145
Thread ID: 5524 Current Counter = 44147
Thread ID: 8572 Current Counter = 44147
Thread ID: 8572 Current Counter = 44149

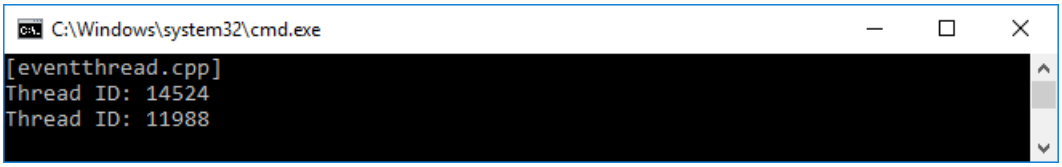

C:\Windows\system32\cmd.exe
Thread ID: 6568 Current Counter = 49992
Thread ID: 6568 Current Counter = 49993
Thread ID: 6568 Current Counter = 49994
Thread ID: 6568 Current Counter = 49995
Thread ID: 6568 Current Counter = 49996
Thread ID: 6568 Current Counter = 49997
Thread ID: 6568 Current Counter = 49998
Thread ID: 6568 Current Counter = 49999
Thread ID: 6568 Current Counter = 50000
Final result = 50000

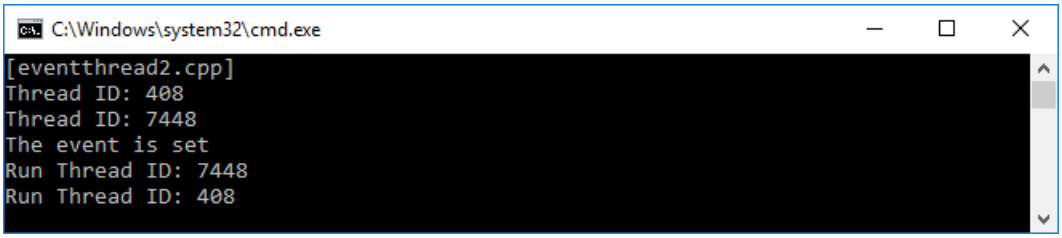

C:\Windows\system32\cmd.exe
[recursivemutex.cpp]
Multiplexer() is called. m_content = 10
Divisor() is called. m_content = 1

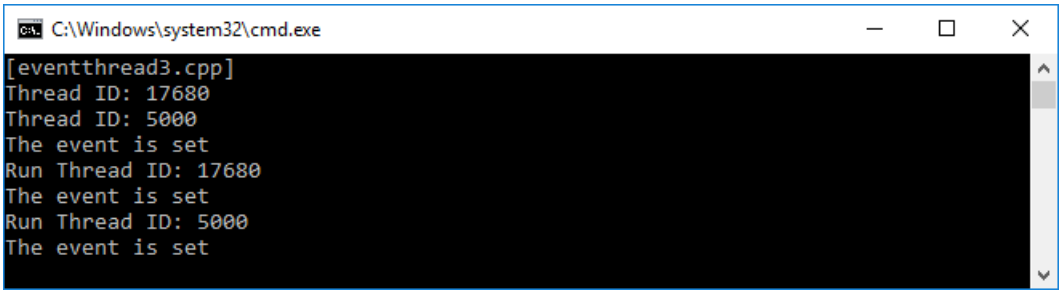

C:\Windows\system32\cmd.exe
[threadhandle.cpp]
threadProc() is run.
The result = 100


C:\Windows\system32\cmd.exe
[threaduniquehandle.cpp]
threadProc() is run.
After running thread
The result = 100

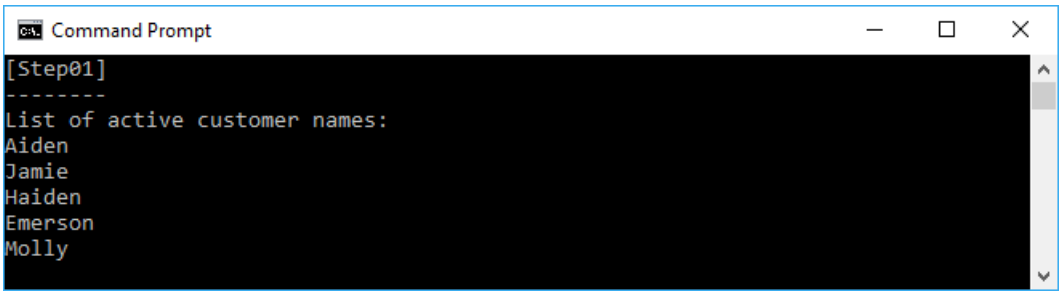

C:\Windows\system32\cmd.exe
[event.cpp]
The event is not signaled
The event is set
The event is signaled
The event is cleared
The event is not signaled

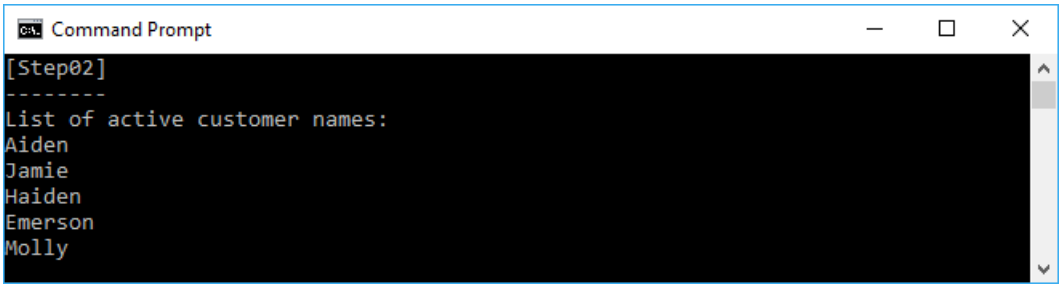

C:\Windows\system32\cmd.exe
[eventthread.cpp]
Thread ID: 14524
Thread ID: 11988

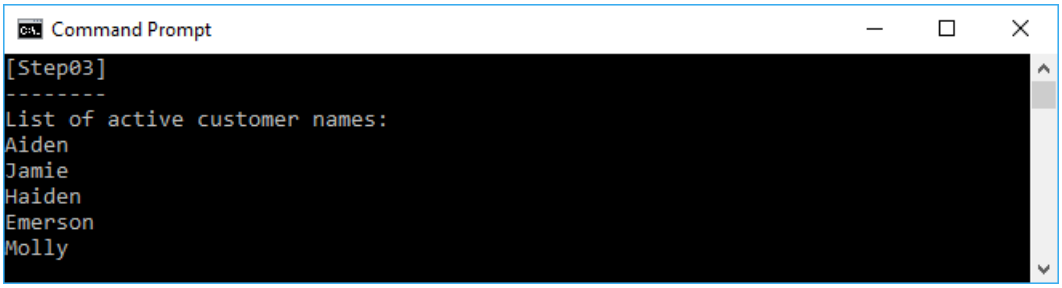

C:\Windows\system32\cmd.exe
[eventthread2.cpp]
Thread ID: 408
Thread ID: 7448
The event is set
Run Thread ID: 7448
Run Thread ID: 408

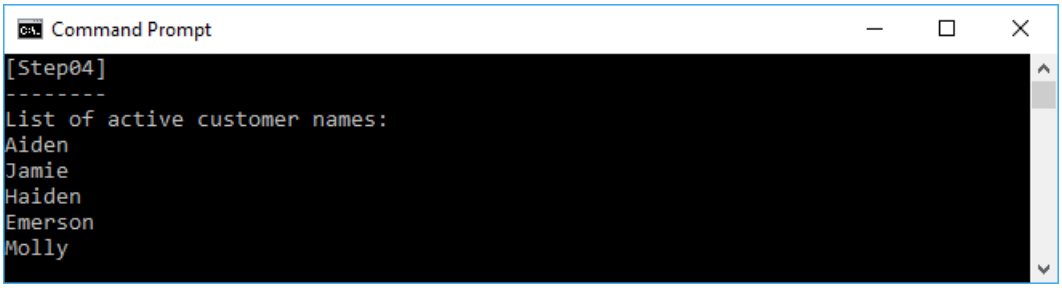

C:\Windows\system32\cmd.exe
[eventthread3.cpp]
Thread ID: 17680
Thread ID: 5000
The event is set
Run Thread ID: 17680
The event is set
Run Thread ID: 5000
The event is set

Chapter 8: Creating and debugging application in functional approach

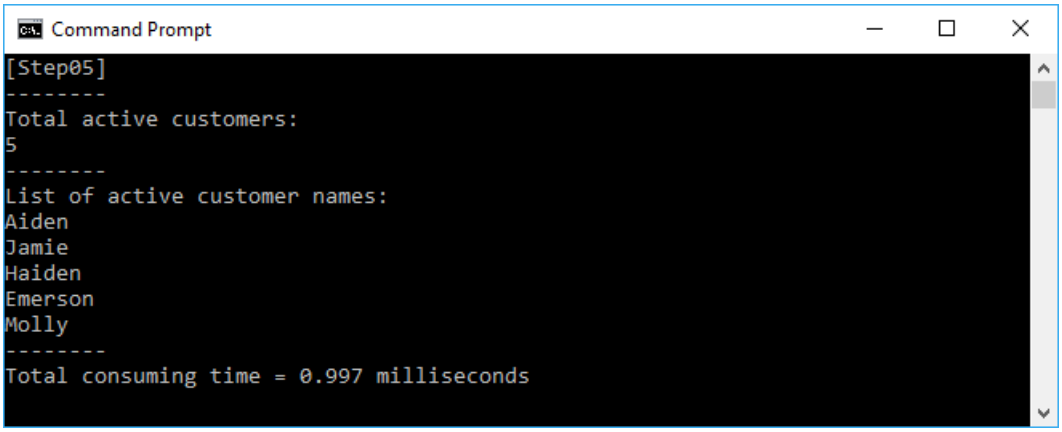







Command Prompt
[Step04]

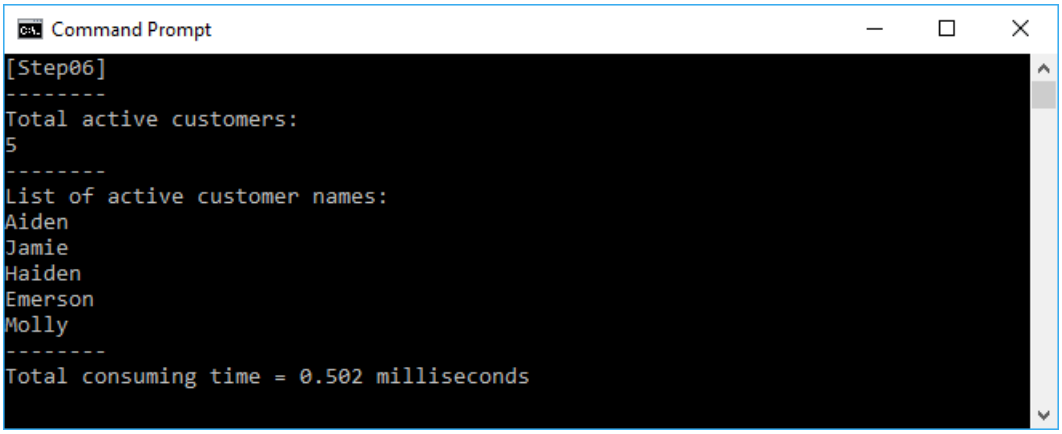
List of active customer names:
Aiden
Jamie
Haiden
Emerson
Molly


Command Prompt
[Step05]

Total active customers:
5

List of active customer names:
Aiden
Jamie
Haiden
Emerson
Molly

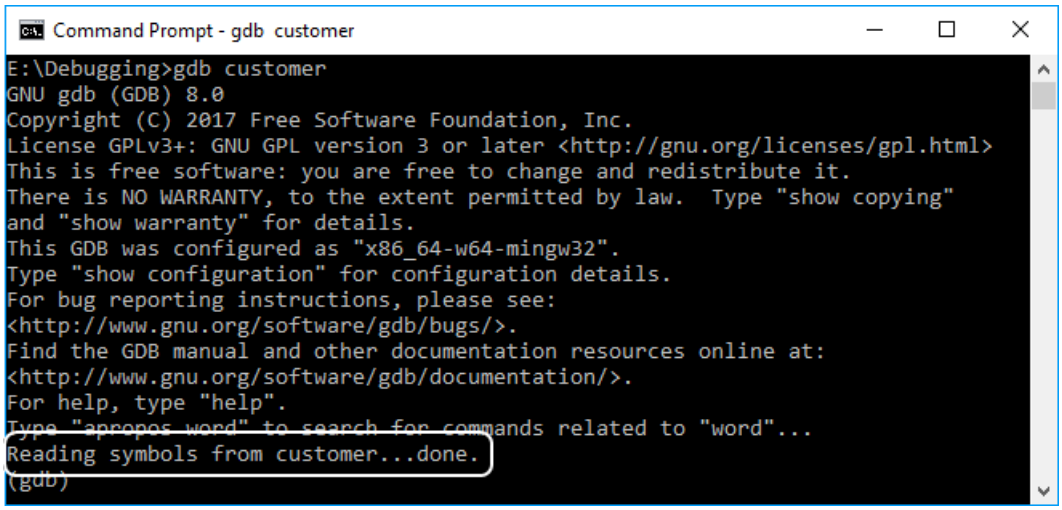
Total consuming time = 0.997 milliseconds

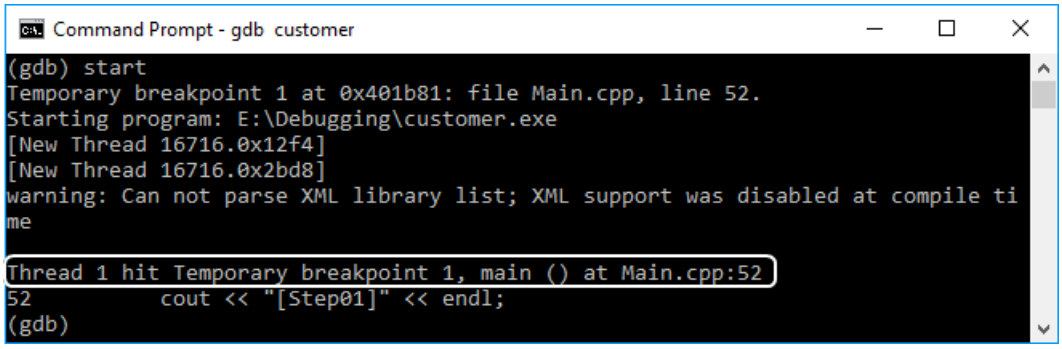

Command Prompt
[Step06]

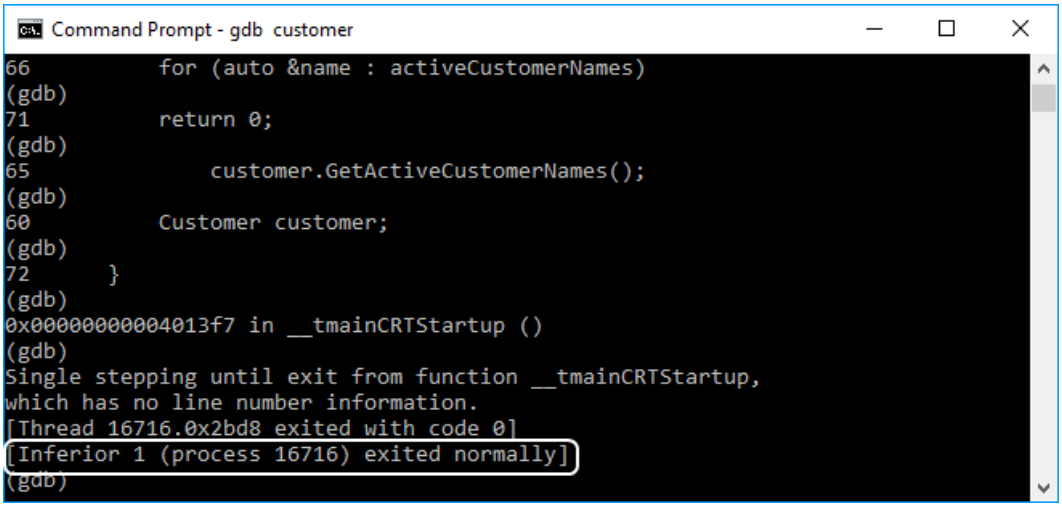
Total active customers:
5

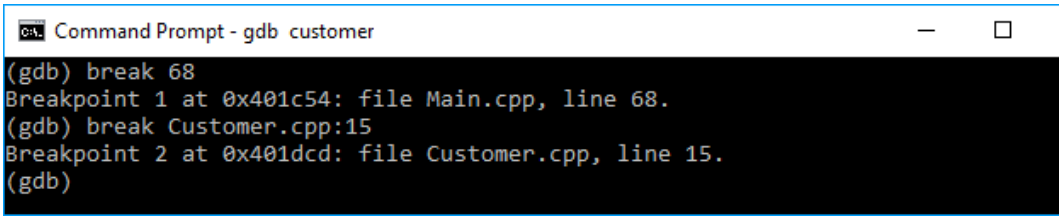
List of active customer names:
Aiden
Jamie
Haiden
Emerson
Molly

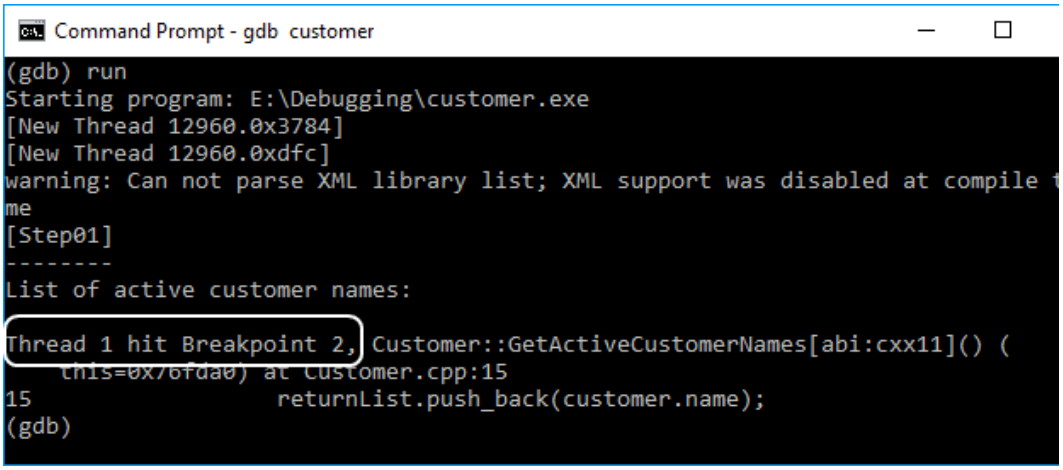
Total consuming time = 0.502 milliseconds


Command Prompt - gdb customer
E:\Debugging>gdb customer
GNU gdb (GDB) 8.0
Copyright (C) 2017 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-w64-mingw32".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from customer...done.
(gdb)

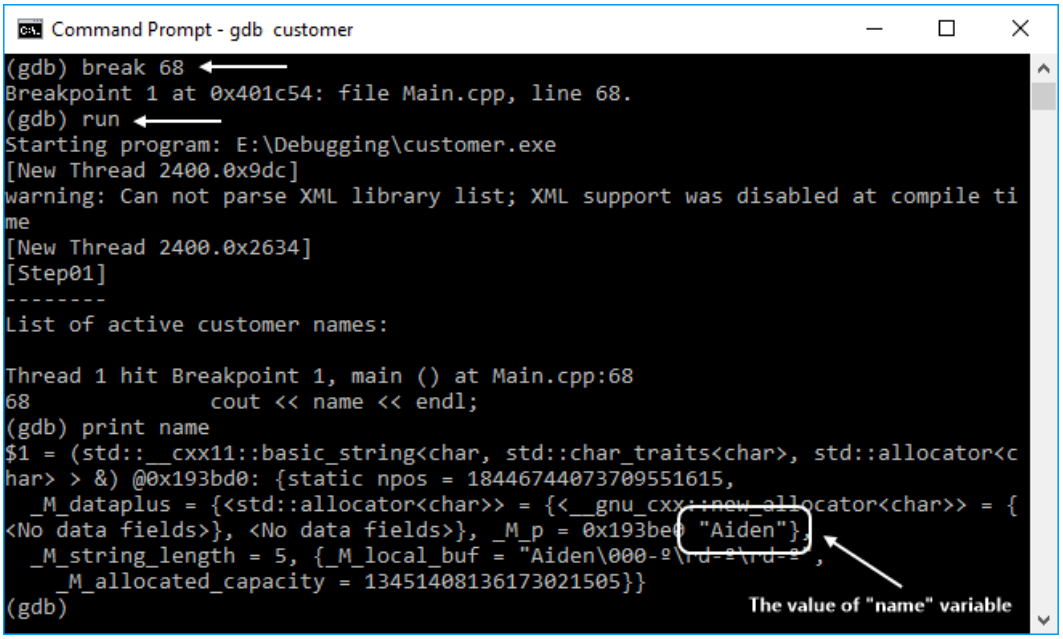

Command Prompt - gdb customer
(gdb) start
Temporary breakpoint 1 at 0x401b81: file Main.cpp, line 52.
Starting program: E:\Debugging\customer.exe
[New Thread 16716.0x12f4]
[New Thread 16716.0x2bd8]
warning: Can not parse XML library list; XML support was disabled at compile ti
me
Thread 1 hit Temporary breakpoint 1, main () at Main.cpp:52
52 cout << "[Step01]" << endl;
(gdb)


Command Prompt - gdb customer
66 for (auto &name : activeCustomerNames)
(gdb)
71 return 0;
(gdb)
65 customer.GetActiveCustomerNames();
(gdb)
60 Customer customer;
(gdb)
72 }
(gdb)
0x00000000004013f7 in __tmainCRTStartup ()
(gdb)
Single stepping until exit from function __tmainCRTStartup,
which has no line number information.
[Thread 16716.0x2bd8 exited with code 0]
[Inferior 1 (process 16716) exited normally]
(gdb)


Command Prompt - gdb customer
(gdb) break 68
Breakpoint 1 at 0x401c54: file Main.cpp, line 68.
(gdb) break Customer.cpp:15
Breakpoint 2 at 0x401dcd: file Customer.cpp, line 15.
(gdb)


Command Prompt - gdb customer
(gdb) run
Starting program: E:\Debugging\customer.exe
[New Thread 12960.0x3784]
[New Thread 12960.0xdfc]
warning: Can not parse XML library list; XML support was disabled at compile t
me
[Step01]

List of active customer names:
Thread 1 hit Breakpoint 2, Customer::GetActiveCustomerNames[abi:cxx11]() (
this=0x76fda0) at Customer.cpp:15
15 returnList.push_back(customer.name);
(gdb)


Command Prompt - gdb customer
(gdb) break 68
Breakpoint 1 at 0x401c54: file Main.cpp, line 68.
(gdb) run
Starting program: E:\Debugging\customer.exe
[New Thread 2400.0x9dc]
warning: Can not parse XML library list; XML support was disabled at compile ti
me
[New Thread 2400.0x2634]
[Step01]

List of active customer names:
Thread 1 hit Breakpoint 1, main () at Main.cpp:68
68 cout << name << endl;
(gdb) print name
$1 = (std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<c
har> > &) @0x193bd0: {static npos = 18446744073709551615,
_M_dataplus = {<std::allocator<char>> = {<__gnu_cxx::new_allocator<char>> = {
<No data fields>}, <No data fields>}, _M_p = 0x193be0 "Aiden"},
_M_string_length = 5, {_M_local_buf = "Aiden\000-º\rd-º\rd-º",
_M_allocated_capacity = 13451408136173021505}}
(gdb)
The value of "name" variable