

An Example of Constructors/Destructors Being Called

```
1 // Fig. 9.15: fig09_15.cpp
2 // Demonstrating the order in which constructors and
3 // destructors are called.
4 #include <iostream>
5 #include "CreateAndDestroy.h" // include CreateAndDestroy class definition
6 using namespace std;
7
8 void create( void ); // prototype
9
10 CreateAndDestroy first( 1, "(global before main)" ); // global object
11
12 int main()
13 {
14     cout << "\nMAIN FUNCTION: EXECUTION BEGINS" << endl;
15     CreateAndDestroy second( 2, "(local automatic in main)" );
16     static CreateAndDestroy third( 3, "(local static in main)" );
17
18     create(); // call function to create objects
19
20     cout << "\nMAIN FUNCTION: EXECUTION RESUMES" << endl;
21     CreateAndDestroy fourth( 4, "(local automatic in main)" );
22     cout << "\nMAIN FUNCTION: EXECUTION ENDS" << endl;
23 } // end main
24
25 // function to create objects
26 void create( void )
27 {
28     cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
29     CreateAndDestroy fifth( 5, "(local automatic in create)" );
30     static CreateAndDestroy sixth( 6, "(local static in create)" );
31     CreateAndDestroy seventh( 7, "(local automatic in create)" );
32     cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
33 } // end function create
```



An Example of Constructors/Destructors Being Called

```
1 // 1. Locations for static data are reserved
2
3 // destructors are called.
4 #include <iostream>
5 #include "CreateAndDestroy.h" // include CreateAndDestroy class definition
6 using namespace std;
7
8 void create( void ); // prototype
9
10 CreateAndDestroy first( 1, "(global before main)" ); // global object
11
12 int main()
13 {
14     cout << "\nMAIN FUNCTION: EXECUTION BEGINS" << endl;
15     CreateAndDestroy second( 2, "(local automatic in main)" );
16     static CreateAndDestroy third( 3, "(local static in main)" );
17
18     create(); // call function to create objects
19
20     cout << "\nMAIN FUNCTION: EXECUTION RESUMES" << endl;
21     CreateAndDestroy fourth( 4, "(local automatic in main)" );
22     cout << "\nMAIN FUNCTION: EXECUTION ENDS" << endl;
23 } // end main
24
25 // function to create objects
26 void create( void )
27 {
28     cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
29     CreateAndDestroy fifth( 5, "(local automatic in create)" );
30     static CreateAndDestroy sixth( 6, "(local static in create)" );
31     CreateAndDestroy seventh( 7, "(local automatic in create)" );
32     cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
33 } // end function create
```

Code

first
third
sixth



An Example of Constructors/Destructors Being Called

```
1 1. Locations for static data are reserved
2 2. Calling constructor for first
3
4 #include <iostream>
5 #include "CreateAndDestroy.h" // include CreateAndDestroy class definition
6 using namespace std;
7
8 void create( void ); // prototype
9
10 CreateAndDestroy first( 1, "(global before main)" ); // global object
11
12 int main()
13 {
14     cout << "\nMAIN FUNCTION: EXECUTION BEGINS" << endl;
15     CreateAndDestroy second( 2, "(local automatic in main)" );
16     static CreateAndDestroy third( 3, "(local static in main)" );
17
18     create(); // call function to create objects
19
20     cout << "\nMAIN FUNCTION: EXECUTION RESUMES" << endl;
21     CreateAndDestroy fourth( 4, "(local automatic in main)" );
22     cout << "\nMAIN FUNCTION: EXECUTION ENDS" << endl;
23 } // end main
24
25 // function to create objects
26 void create( void )
27 {
28     cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
29     CreateAndDestroy fifth( 5, "(local automatic in create)" );
30     static CreateAndDestroy sixth( 6, "(local static in create)" );
31     CreateAndDestroy seventh( 7, "(local automatic in create)" );
32     cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
33 } // end function create
```

Code

first
third
sixth



An Example of Constructors/Destructors Being Called

```
1 1. Locations for static data are reserved
2 2. Calling constructor for first
3 3. Pushing main's ARI to stack (locations for local variables and
4 # parameters are reserved)
5 #
6 #  
7  
8 void create( void ); // prototype
9  
10 CreateAndDestroy first( 1, "(global before main)" ); // global object
11  
12 int main()
13 {
14     cout << "\nMAIN FUNCTION: EXECUTION BEGINS" << endl;
15     CreateAndDestroy second( 2, "(local automatic in main)" );
16     static CreateAndDestroy third( 3, "(local static in main)" );
17  
18     create(); // call function to create objects
19  
20     cout << "\nMAIN FUNCTION: EXECUTION RESUMES" << endl;
21     CreateAndDestroy fourth( 4, "(local automatic in main)" );
22     cout << "\nMAIN FUNCTION: EXECUTION ENDS" << endl;
23 } // end main
24  
25 // function to create objects
26 void create( void )
27 {
28     cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
29     CreateAndDestroy fifth( 5, "(local automatic in create)" );
30     static CreateAndDestroy sixth( 6, "(local static in create)" );
31     CreateAndDestroy seventh( 7, "(local automatic in create)" );
32     cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
33 } // end function create
```

definition

Code

first
third
sixth

For main

fourth
second



An Example of Constructors/Destructors Being Called

```
1 1. Locations for static data are reserved
2 2. Calling constructor for first
3 3. Pushing main's ARI to stack (locations for local variables and
4 # parameters are reserved)
5 # 4. Calling constructor for second
6 U
7
8 void create( void ); // prototype
9
10 CreateAndDestroy first( 1, "(global before main)" ); // global object
11
12 int main()
13 {
14     cout << "\nMAIN FUNCTION: EXECUTION BEGINS" << endl;
15     CreateAndDestroy second( 2, "(local automatic in main)" );
16     static CreateAndDestroy third( 3, "(local static in main)" );
17
18     create(); // call function to create objects
19
20     cout << "\nMAIN FUNCTION: EXECUTION RESUMES" << endl;
21     CreateAndDestroy fourth( 4, "(local automatic in main)" );
22     cout << "\nMAIN FUNCTION: EXECUTION ENDS" << endl;
23 } // end main
24
25 // function to create objects
26 void create( void )
27 {
28     cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
29     CreateAndDestroy fifth( 5, "(local automatic in create)" );
30     static CreateAndDestroy sixth( 6, "(local static in create)" );
31     CreateAndDestroy seventh( 7, "(local automatic in create)" );
32     cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
33 } // end function create
```

definition

Code

first
third
sixth

For main

fourth
second



An Example of Constructors/Destructors Being Called

```
1 1. Locations for static data are reserved
2 2. Calling constructor for first
3 3. Pushing main's ARI to stack (locations for local variables and
4 # parameters are reserved)
5 # 4. Calling constructor for second
6 U 5. Calling constructor for third
7 V
8
9
10 CreateAndDestroy first( 1, "(global before main)" ); // global object
11
12 int main()
13 {
14     cout << "\nMAIN FUNCTION: EXECUTION BEGINS" << endl;
15     CreateAndDestroy second( 2, "(local automatic in main)" );
16     static CreateAndDestroy third( 3, "(local static in main)" );
17
18     create(); // call function to create objects
19
20     cout << "\nMAIN FUNCTION: EXECUTION RESUMES" << endl;
21     CreateAndDestroy fourth( 4, "(local automatic in main)" );
22     cout << "\nMAIN FUNCTION: EXECUTION ENDS" << endl;
23 } // end main
24
25 // function to create objects
26 void create( void )
27 {
28     cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
29     CreateAndDestroy fifth( 5, "(local automatic in create)" );
30     static CreateAndDestroy sixth( 6, "(local static in create)" );
31     CreateAndDestroy seventh( 7, "(local automatic in create)" );
32     cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
33 } // end function create
```

definition

Code

first
third
sixth

For main

fourth
second



An Example of Constructors/Destructors Being Called

```
1 1. Locations for static data are reserved
2 2. Calling constructor for first
3 3. Pushing main's ARI to stack (locations for local variables and
4 # parameters are reserved)
5 # 
6 v
7 v
8 v
9 v
10 C
11 int main()
12 {
13     cout << "\nMAIN FUNCTION: EXECUTION BEGINS" << endl;
14     CreateAndDestroy second( 2, "(local automatic in main)" );
15     static CreateAndDestroy third( 3, "(local static in main)" );
16
17     create(); // call function to create objects
18
19     cout << "\nMAIN FUNCTION: EXECUTION RESUMES" << endl;
20     CreateAndDestroy fourth( 4, "(local automatic in main)" );
21     cout << "\nMAIN FUNCTION: EXECUTION ENDS" << endl;
22 } // end main
23
24 // function to create objects
25 void create( void )
26 {
27     cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
28     CreateAndDestroy fifth( 5, "(local automatic in create)" );
29     static CreateAndDestroy sixth( 6, "(local static in create)" );
30     CreateAndDestroy seventh( 7, "(local automatic in create)" );
31     cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
32 } // end function create
```

definition

object

Code

first
third
sixth

For create

seventh
fifth

For main

fourth
second



An Example of Constructors/Destructors Being Called

```
1 1. Locations for static data are reserved
2 2. Calling constructor for first
3 3. Pushing main's ARI to stack (locations for local variables and
4 # parameters are reserved)
5 # 4. Calling constructor for second
6 U 5. Calling constructor for third
7 v 6. Pushing create's ARI to stack (locations for local variables
8 C and parameters are reserved)
9 O 7. Calling constructor for fifth
10 C
11 i
12 {
13
14 cout << "\nMAIN FUNCTION: EXECUTION BEGINS" << endl;
15 CreateAndDestroy second( 2, "(local automatic in main)" );
16 static CreateAndDestroy third( 3, "(local static in main)" );
17
18 create(); // call function to create objects
19
20 cout << "\nMAIN FUNCTION: EXECUTION RESUMES" << endl;
21 CreateAndDestroy fourth( 4, "(local automatic in main)" );
22 cout << "\nMAIN FUNCTION: EXECUTION ENDS" << endl;
23 } // end main
24
25 // function to create objects
26 void create( void )
27 {
28 cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
29 CreateAndDestroy fifth( 5, "(local automatic in create)" );
30 static CreateAndDestroy sixth( 6, "(local static in create)" );
31 CreateAndDestroy seventh( 7, "(local automatic in create)" );
32 cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
33 } // end function create
```

definition

object

Code

first
third
sixth

For create

seventh

fifth

For main

fourth

second



An Example of Constructors/Destructors Being Called

```
1 1. Locations for static data are reserved
2 2. Calling constructor for first
3 3. Pushing main's ARI to stack (locations for local variables and
4 # parameters are reserved)
5 # 4. Calling constructor for second
6 U 5. Calling constructor for third
7 V 6. Pushing create's ARI to stack (locations for local variables
8 C and parameters are reserved)
9 C 7. Calling constructor for fifth
10 C 8. Calling constructor for sixth
11 C
12 C
13 C
14 cout << "\nMAIN FUNCTION: EXECUTION BEGINS" << endl;
15 CreateAndDestroy second( 2, "(local automatic in main)" );
16 static CreateAndDestroy third( 3, "(local static in main)" );
17
18 create(); // call function to create objects
19
20 cout << "\nMAIN FUNCTION: EXECUTION RESUMES" << endl;
21 CreateAndDestroy fourth( 4, "(local automatic in main)" );
22 cout << "\nMAIN FUNCTION: EXECUTION ENDS" << endl;
23 } // end main
24
25 // function to create objects
26 void create( void )
27 {
28 cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
29 CreateAndDestroy fifth( 5, "(local automatic in create)" );
30 static CreateAndDestroy sixth( 6, "(local static in create)" );
31 CreateAndDestroy seventh( 7, "(local automatic in create)" );
32 cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
33 } // end function create
```

definition

object

Code

first
third
sixth

For create

seventh

fifth

For main

fourth

second



An Example of Constructors/Destructors Being Called

```
1 1. Locations for static data are reserved
2 2. Calling constructor for first
3 3. Pushing main's ARI to stack (locations for local variables and
4 # parameters are reserved)
5 # 4. Calling constructor for second
6 U 5. Calling constructor for third
7 V 6. Pushing create's ARI to stack (locations for local variables
8 C and parameters are reserved)
9 O 7. Calling constructor for fifth
10 C 8. Calling constructor for sixth
11 C 9. Calling constructor for seventh
12 C
13 C
14 C
15 CreateAndDestroy second( 2, "(local automatic in main)" );
16 static CreateAndDestroy third( 3, "(local static in main)" );
17
18 create(); // call function to create objects
19
20 cout << "\nMAIN FUNCTION: EXECUTION RESUMES" << endl;
21 CreateAndDestroy fourth( 4, "(local automatic in main)" );
22 cout << "\nMAIN FUNCTION: EXECUTION ENDS" << endl;
23 } // end main
24
25 // function to create objects
26 void create( void )
27 {
28 cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
29 CreateAndDestroy fifth( 5, "(local automatic in create)" );
30 static CreateAndDestroy sixth( 6, "(local static in create)" );
31 CreateAndDestroy seventh( 7, "(local automatic in create)" );
32 cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
33 } // end function create
```

definition

object

Code

first
third
sixth

For create

seventh
fifth

For main

fourth
second



An Example of Constructors/Destructors Being Called

```
1 1. Locations for static data are reserved
2 2. Calling constructor for first
3 3. Pushing main's ARI to stack (locations for local variables and
4 # parameters are reserved)
5 # 4. Calling constructor for second
6 # 5. Calling constructor for third
7 v 6. Pushing create's ARI to stack (locations for local variables
8 C and parameters are reserved)
9 C 7. Calling constructor for fifth
10 C 8. Calling constructor for sixth
11 C 9. Calling constructor for seventh
12 C 10. Calling destructor for seventh
13 C 11. Calling destructor for fifth
14
15 create(); // call function to create objects
16
17 cout << "\nMAIN FUNCTION: EXECUTION RESUMES" << endl;
18 CreateAndDestroy fourth( 4, "(local automatic in main)" );
19 cout << "\nMAIN FUNCTION: EXECUTION ENDS" << endl;
20 } // end main
21
22 // function to create objects
23 void create( void )
24 {
25   cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
26   CreateAndDestroy fifth( 5, "(local automatic in create)" );
27   static CreateAndDestroy sixth( 6, "(local static in create)" );
28   CreateAndDestroy seventh( 7, "(local automatic in create)" );
29   cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
30 } // end function create
31
32
33 }
```

definition

object

Code

first
third
sixth

For create

seventh
fifth

For main

fourth
second



An Example of Constructors/Destructors Being Called

```
1 1. Locations for static data are reserved
2 2. Calling constructor for first
3 3. Pushing main's ARI to stack (locations for local variables and
4 # parameters are reserved)
5 # 4. Calling constructor for second
6 # 5. Calling constructor for third
7 v 6. Pushing create's ARI to stack (locations for local variables
8 C and parameters are reserved)
9 C 7. Calling constructor for fifth
10 C 8. Calling constructor for sixth
11 C 9. Calling constructor for seventh
12 C 10. Calling destructor for seventh
13 C 11. Calling destructor for fifth
14 C 12. Popping create's ARI from stack
15
16     cout << "\nMAIN FUNCTION: EXECUTION RESUMES" << endl;
17     CreateAndDestroy fourth( 4, "(local automatic in main)" );
18     cout << "\nMAIN FUNCTION: EXECUTION ENDS" << endl;
19 } // end main
20
21 // function to create objects
22 void create( void )
23 {
24     cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
25     CreateAndDestroy fifth( 5, "(local automatic in create)" );
26     static CreateAndDestroy sixth( 6, "(local static in create)" );
27     CreateAndDestroy seventh( 7, "(local automatic in create)" );
28     cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
29 } // end function create
```

definition

object

Code

first
third
sixth

For main

fourth
second



An Example of Constructors/Destructors Being Called

```
1 1. Locations for static data are reserved
2 2. Calling constructor for first
3 3. Pushing main's ARI to stack (locations for local variables and
4 # parameters are reserved)
5 # 4. Calling constructor for second
6 # 5. Calling constructor for third
7 v 6. Pushing create's ARI to stack (locations for local variables
8 C and parameters are reserved)
9 C 7. Calling constructor for fifth
10 C 8. Calling constructor for sixth
11 C 9. Calling constructor for seventh
12 C 10. Calling destructor for seventh
13 C 11. Calling destructor for fifth
14 C 12. Popping create's ARI from stack
15 C 13. Calling constructor for fourth
16
17 cout << "\nMAIN FUNCTION: EXECUTION RESUMES" << endl;
18 CreateAndDestroy fourth( 4, "(local automatic in main)" );
19 cout << "\nMAIN FUNCTION: EXECUTION ENDS" << endl;
20 } // end main
21
22 // function to create objects
23 void create( void )
24 {
25     cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
26     CreateAndDestroy fifth( 5, "(local automatic in create)" );
27     static CreateAndDestroy sixth( 6, "(local static in create)" );
28     CreateAndDestroy seventh( 7, "(local automatic in create)" );
29     cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
30 } // end function create
```

definition

object

Code

first
third
sixth

For main

fourth
second



An Example of Constructors/Destructors Being Called

```
1 // main
2
3 #include <iostream>
4 #include "CreateAndDestroy.h"
5
6 using namespace std;
7
8 class CreateAndDestroy
9 {
10 public:
11     CreateAndDestroy( int i, const string& s ) : index( i ), str( s )
12     {
13         cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
14         cout << "index = " << index << ", str = " << str << endl;
15     }
16
17     ~CreateAndDestroy()
18     {
19         cout << "DESTRUCT FUNCTION: EXECUTION ENDS" << endl;
20     }
21
22     void print() const
23     {
24         cout << "index = " << index << ", str = " << str << endl;
25     }
26 };
27
28
29 CreateAndDestroy first( 1, "(local automatic in main)" );
30 static CreateAndDestroy second( 2, "(local static in main)" );
31 CreateAndDestroy third( 3, "(local automatic in main)" );
32 CreateAndDestroy fourth( 4, "(local static in main)" );
33 CreateAndDestroy fifth( 5, "(local automatic in create)" );
34 static CreateAndDestroy sixth( 6, "(local static in create)" );
35 CreateAndDestroy seventh( 7, "(local automatic in create)" );
36
37
38 void main()
39 {
40     cout << "main() starts" << endl;
41
42     CreateAndDestroy( 8, "(local automatic in main)" );
43     cout << "main() ends" << endl;
44 }
```

definition

object

Copyright by Yi-Ping Yu

Code

first
third
sixth

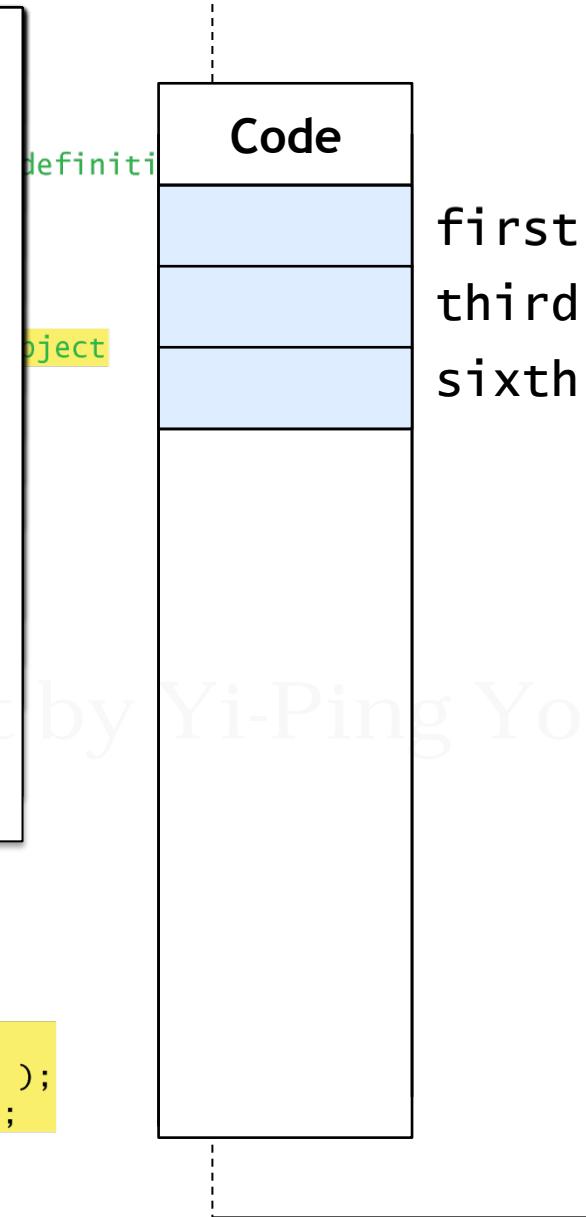
For main

fourth
second



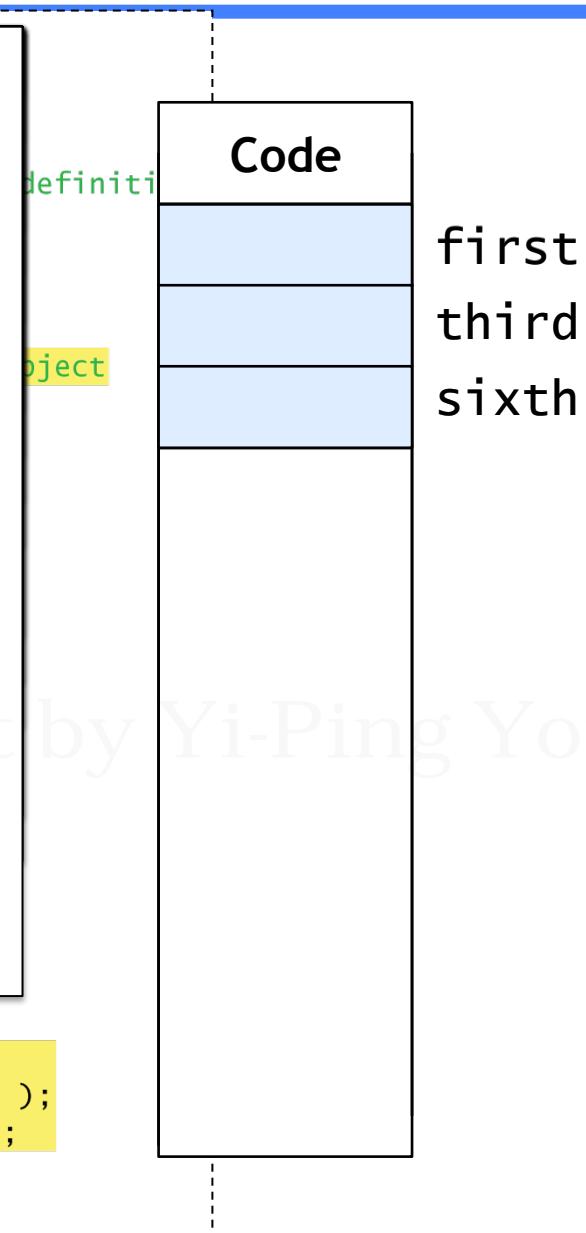
An Example of Constructors/Destructors Being Called

```
1 1. Locations for static data are reserved
2 2. Calling constructor for first
3 3. Pushing main's ARI to stack (locations for local variables and
4 # parameters are reserved)
5 # 4. Calling constructor for second
6 # 5. Calling constructor for third
7 # 6. Pushing create's ARI to stack (locations for local variables
8 # and parameters are reserved)
9 # 7. Calling constructor for fifth
10 # 8. Calling constructor for sixth
11 # 9. Calling constructor for seventh
12 # 10. Calling destructor for seventh
13 # 11. Calling destructor for fifth
14 # 12. Popping create's ARI from stack
15 # 13. Calling constructor for fourth
16 # 14. Calling destructor for fourth
17 # 15. Calling destructor for second
18 # 16. Popping main's ARI from stack
19
20
21
22
23
24
25 // function to create objects
26 void create( void )
27 {
28     cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
29     CreateAndDestroy fifth( 5, "(local automatic in create)" );
30     static CreateAndDestroy sixth( 6, "(local static in create)" );
31     CreateAndDestroy seventh( 7, "(local automatic in create)" );
32     cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
33 } // end function create
```



An Example of Constructors/Destructors Being Called

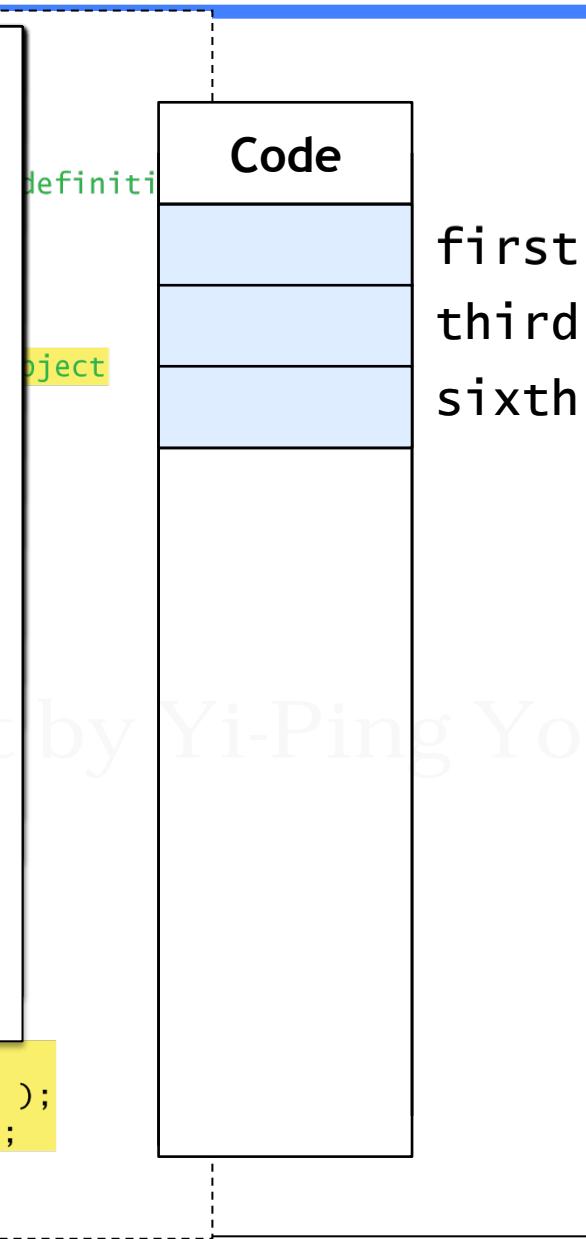
```
1 1. Locations for static data are reserved
2 2. Calling constructor for first
3 3. Pushing main's ARI to stack (locations for local variables and
4 # parameters are reserved)
5 # 4. Calling constructor for second
6 # 5. Calling constructor for third
7 # 6. Pushing create's ARI to stack (locations for local variables
8 # and parameters are reserved)
9 C 7. Calling constructor for fifth
10 C 8. Calling constructor for sixth
11 C 9. Calling constructor for seventh
12 C 10. Calling destructor for seventh
13 C 11. Calling destructor for fifth
14 C 12. Popping create's ARI from stack
15 C 13. Calling constructor for fourth
16 C 14. Calling destructor for fourth
17 C 15. Calling destructor for second
18 C 16. Popping main's ARI from stack
19 C 17. Calling destructor for sixth
20 C 18. Calling destructor for third
21 C 19. Calling destructor for first
22 }
23 cout << "\nCREATE FUNCTION: EXECUTION BEGINS" << endl;
24 CreateAndDestroy fifth( 5, "(local automatic in create)" );
25 static CreateAndDestroy sixth( 6, "(local static in create)" );
26 CreateAndDestroy seventh( 7, "(local automatic in create)" );
27 cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
28 } // end function create
```



An Example of Constructors/Destructors Being Called

```
1 1. Locations for static data are reserved
2 2. Calling constructor for first
3 3. Pushing main's ARI to stack (locations for local variables and
4 # parameters are reserved)
5 # 4. Calling constructor for second
6 # 5. Calling constructor for third
7 # 6. Pushing create's ARI to stack (locations for local variables
8 # and parameters are reserved)
9 # 7. Calling constructor for fifth
10 # 8. Calling constructor for sixth
11 # 9. Calling constructor for seventh
12 # 10. Calling destructor for seventh
13 # 11. Calling destructor for fifth
14 # 12. Popping create's ARI from stack
15 # 13. Calling constructor for fourth
16 # 14. Calling destructor for fourth
17 # 15. Calling destructor for second
18 # 16. Popping main's ARI from stack
19 # 17. Calling destructor for sixth
20 # 18. Calling destructor for third
21 # 19. Calling destructor for first
22 # 20. Program terminates
```

```
29 CreateAndDestroy fifth( 5, "(local automatic in create)" );
30 static CreateAndDestroy sixth( 6, "(local static in create)" );
31 CreateAndDestroy seventh( 7, "(local automatic in create)" );
32 cout << "\nCREATE FUNCTION: EXECUTION ENDS" << endl;
33 } // end function create
```



Memberwise Assignment: An Example

```
#include "time.h"

int main() {
    Time t1(3,10,10,10);
    Time t2(5);
    t2 = t1;

    return 0;
}
```

Code

Copyright by Yi-Ping You



Memberwise Assignment: An Example

```
#include "time.h"

int main() {
    Time t1(3,10,10,10);
    Time t2(5);
    t2 = t1;

    return 0;
}
```

Code

For main

t2.hourHistory
t2.maxHourHistory
t2.numHourHistory
t2.hour
t2.minute
t2.second

t1.hourHistory
t1.maxHourHistory
t1.numHourHistory
t1.hour
t1.minute
t1.second

t2

t1

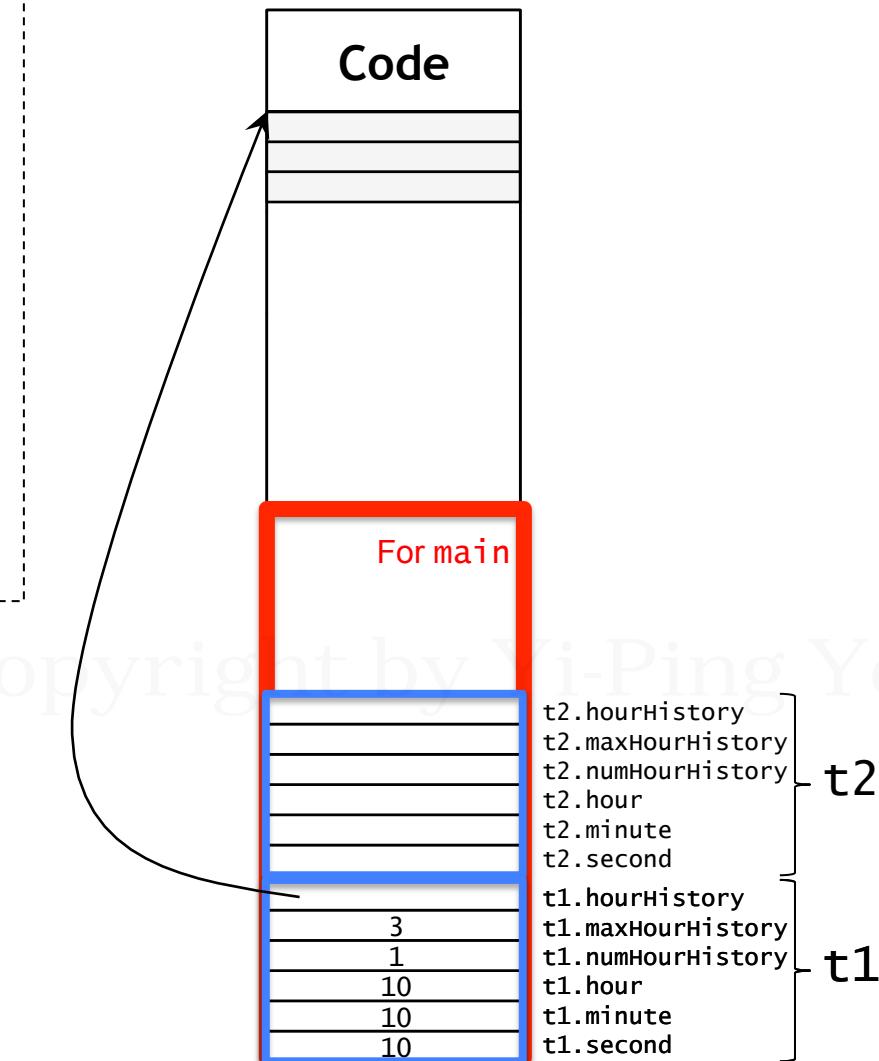


Memberwise Assignment: An Example

```
#include "time.h"

int main() {
    Time t1(3,10,10,10);
    Time t2(5);
    t2 = t1;

    return 0;
}
```

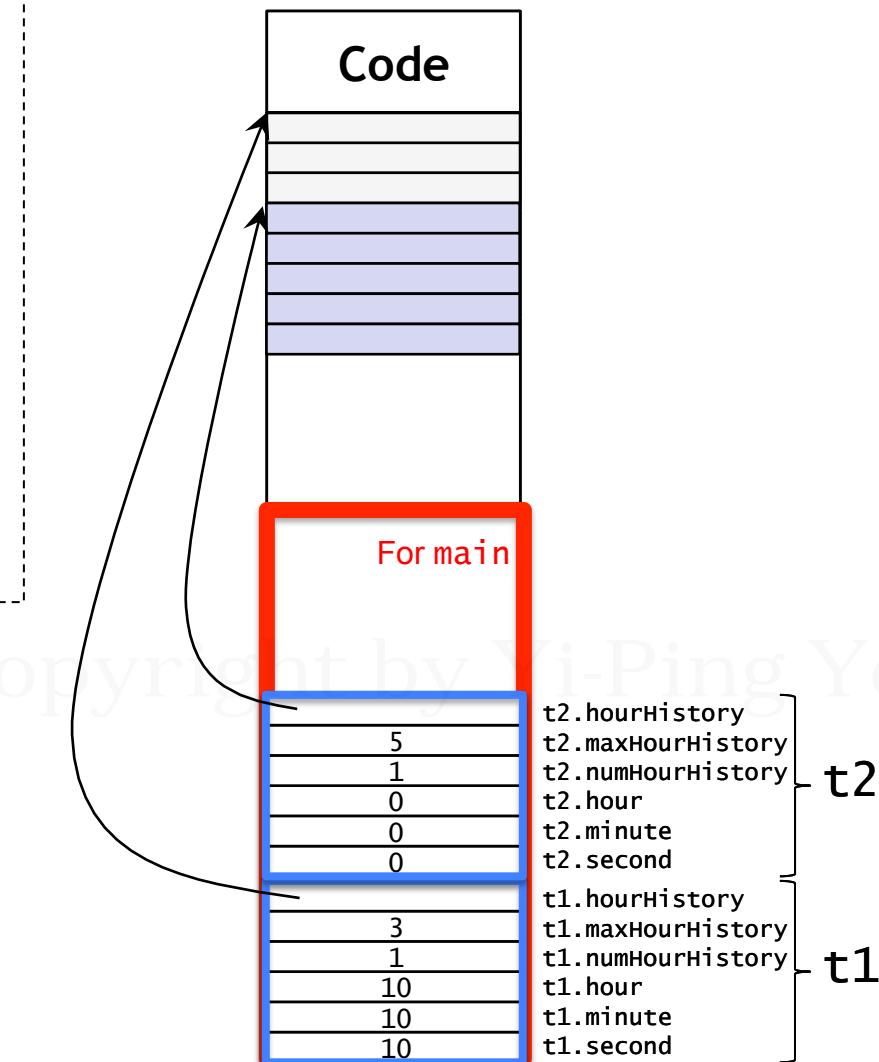


Memberwise Assignment: An Example

```
#include "time.h"

int main() {
    Time t1(3,10,10,10);
    Time t2(5);
    t2 = t1;

    return 0;
}
```

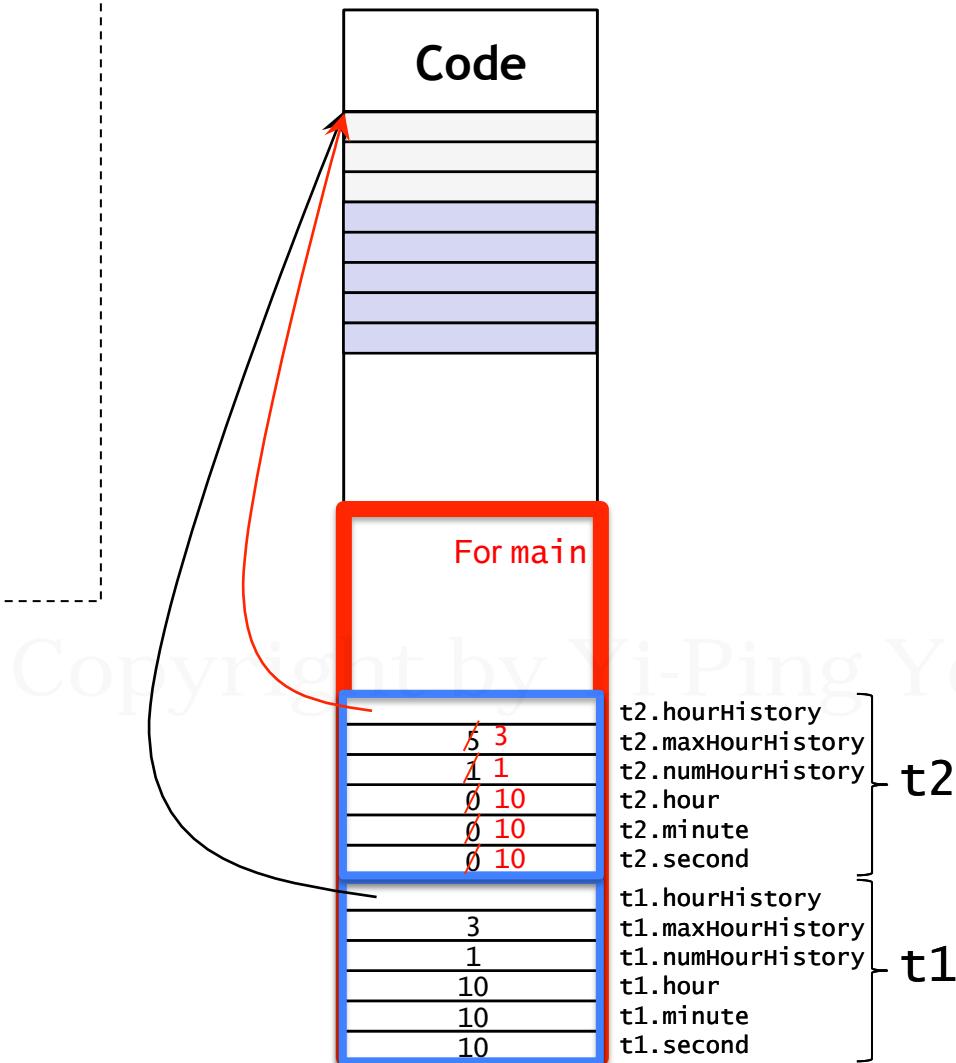


Memberwise Assignment: An Example

```
#include "time.h"

int main() {
    Time t1(3,10,10,10);
    Time t2(5);
    t2 = t1;

    return 0;
}
```

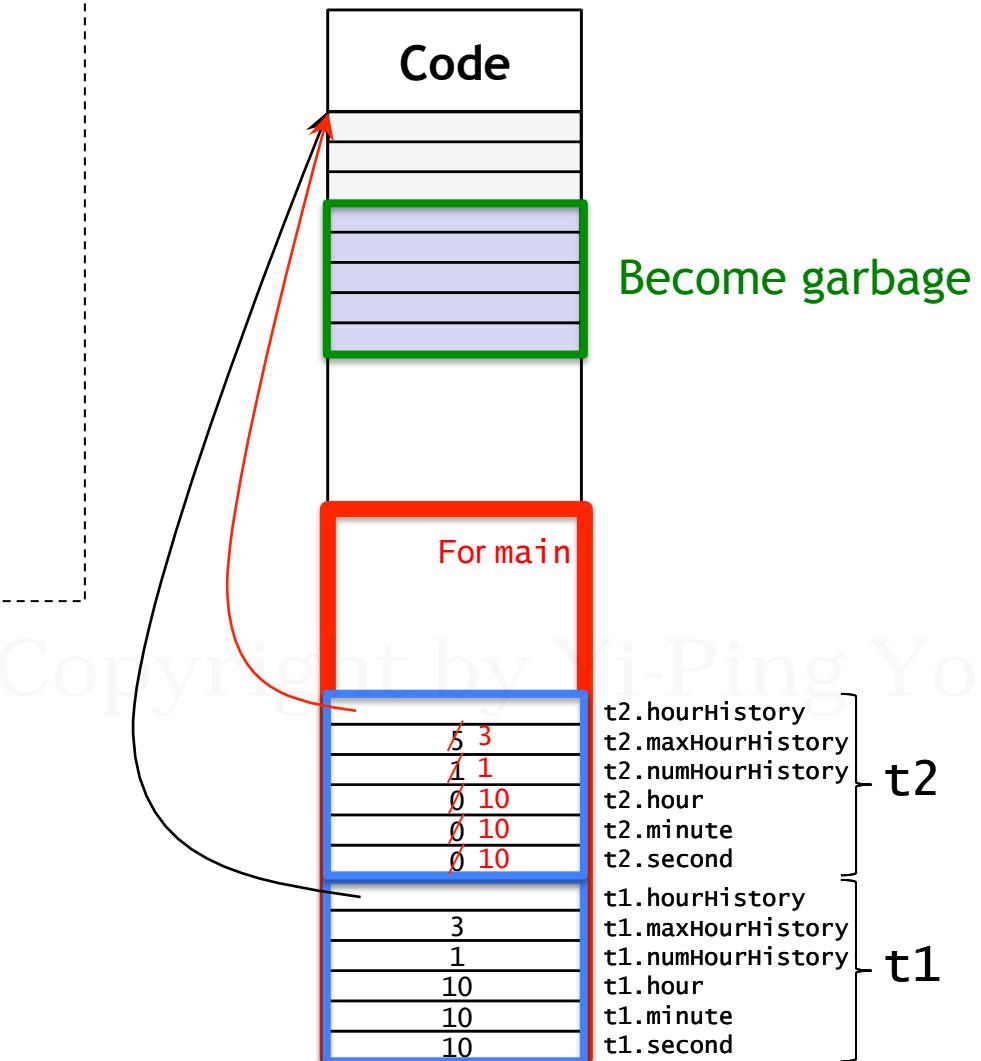


Memberwise Assignment: An Example

```
#include "time.h"

int main() {
    Time t1(3,10,10,10);
    Time t2(5);
    t2 = t1;

    return 0;
}
```



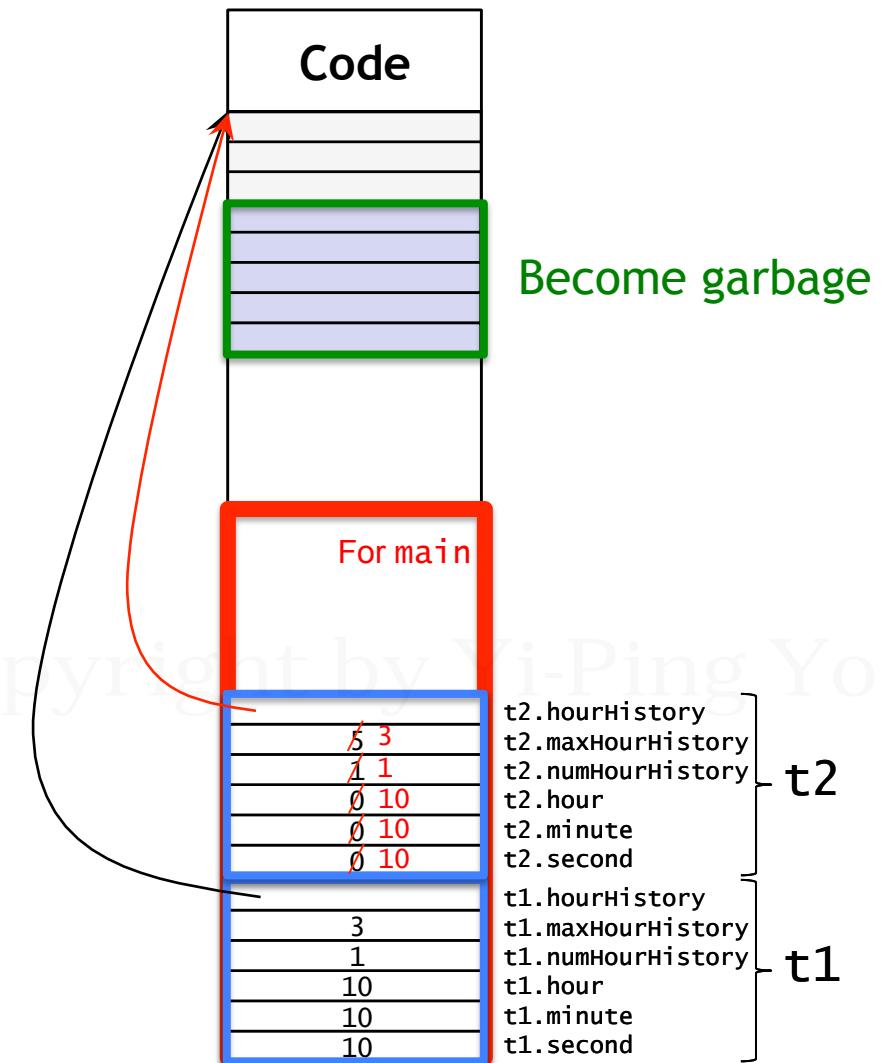
Memberwise Assignment: An Example

```
#include "time.h"

int main() {
    Time t1(3,10,10,10);
    Time t2(5);
    t2 = t1;

    return 0;
}
```

```
time.h
Time::~Time() {
    delete [] hourHistory;
}
```



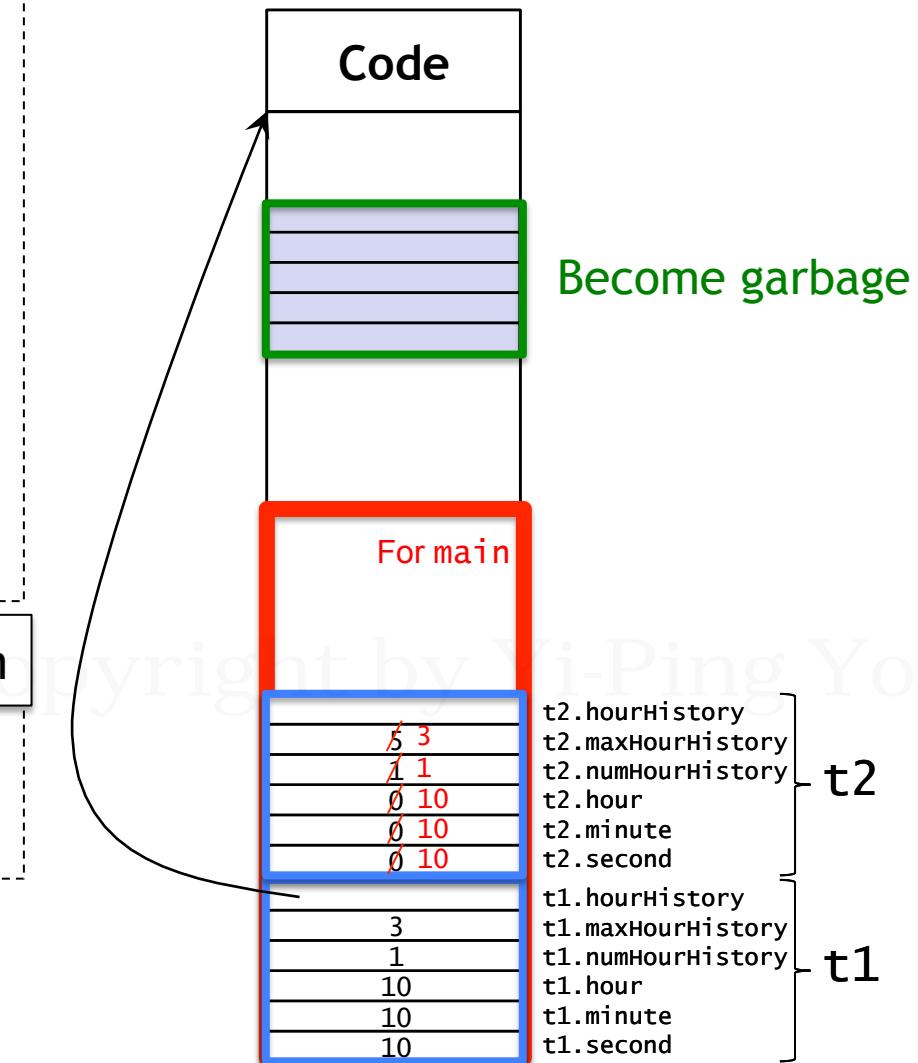
Memberwise Assignment: An Example

```
#include "time.h"

int main() {
    Time t1(3,10,10,10);
    Time t2(5);
    t2 = t1;

    return 0;
}
```

```
time.h
Time::~Time() {
    delete [] hourHistory;
}
```



Memberwise Assignment: An Example

```
#include "time.h"

int main() {
    Time t1(3,10,10,10);
    Time t2(5);
    t2 = t1;

    return 0;
}
```

```
time.h
Time::~Time() {
    delete [] hourHistory;
}
```

Code

Double delete

Become garbage

For main

t2.hourHistory
t2.maxHourHistory
t2.numHourHistory
t2.hour
t2.minute
t2.second
t1.hourHistory
t1.maxHourHistory
t1.numHourHistory
t1.hour
t1.minute
t1.second



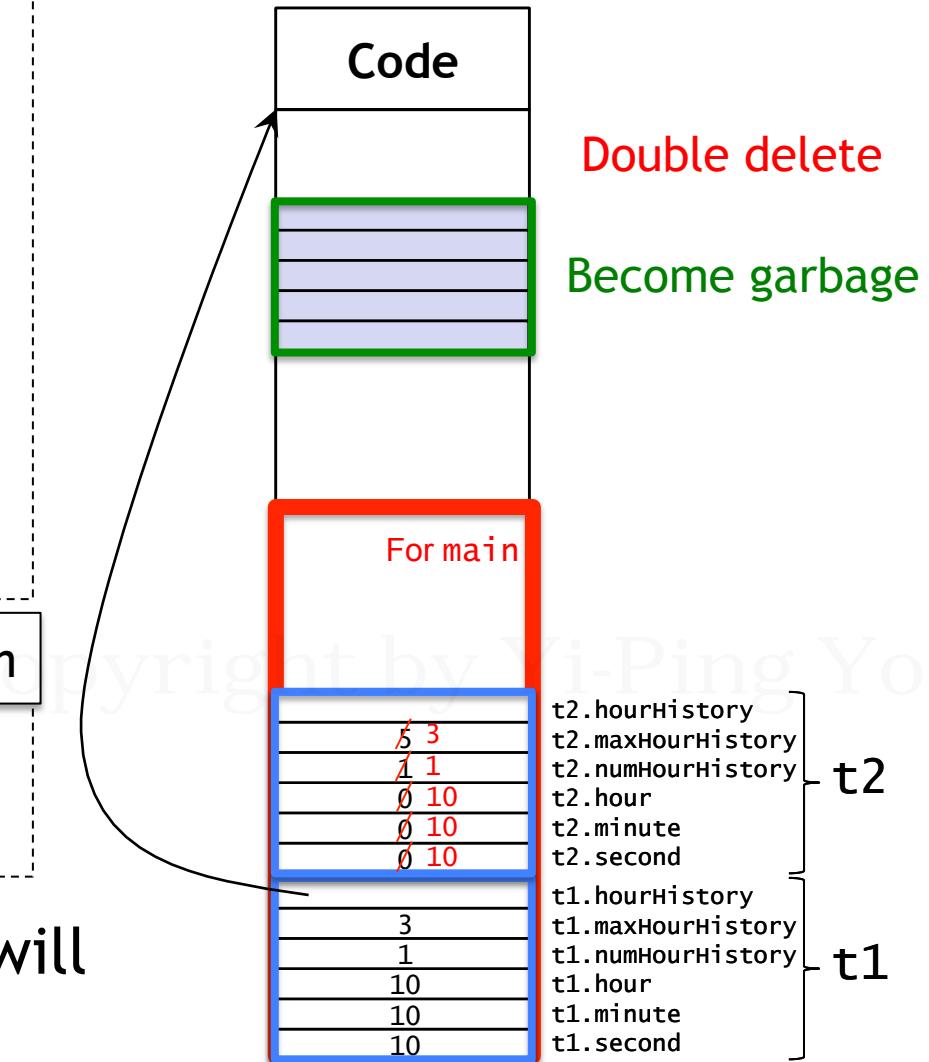
Memberwise Assignment: An Example

```
#include "time.h"

int main() {
    Time t1(3,10,10,10);
    Time t2(5);
    t2 = t1;

    return 0;
}
```

```
time.h
Time::~Time() {
    delete [] hourHistory;
}
```



- Solutions to this problem will be discussed in Chapter 11 (Operator Overloading)

