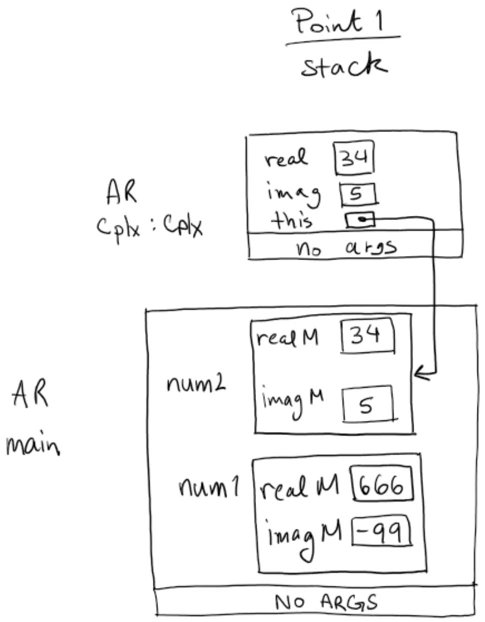
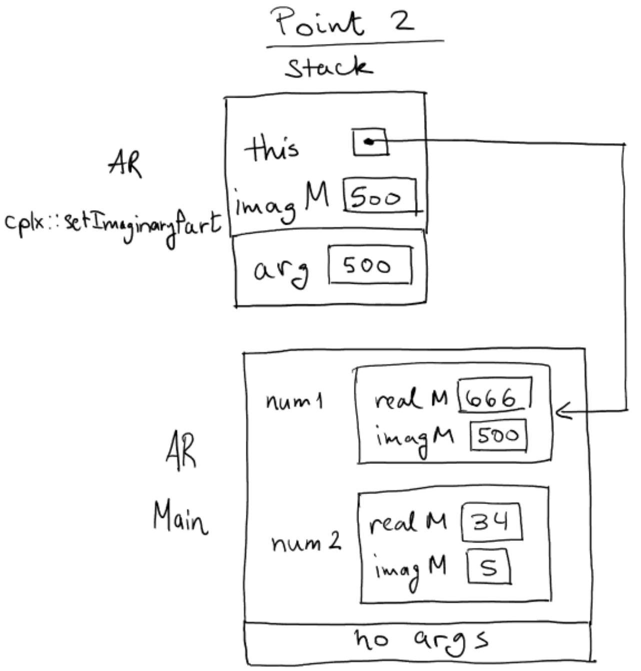
Course: Programming Fundamental – ENSF 337
Lab #: Lab 7
Instructor: M. Moussavi
Student Name: Aarushi Roy Choudhury
Lab Section: B01
Date submitted: Nov, 18 2021

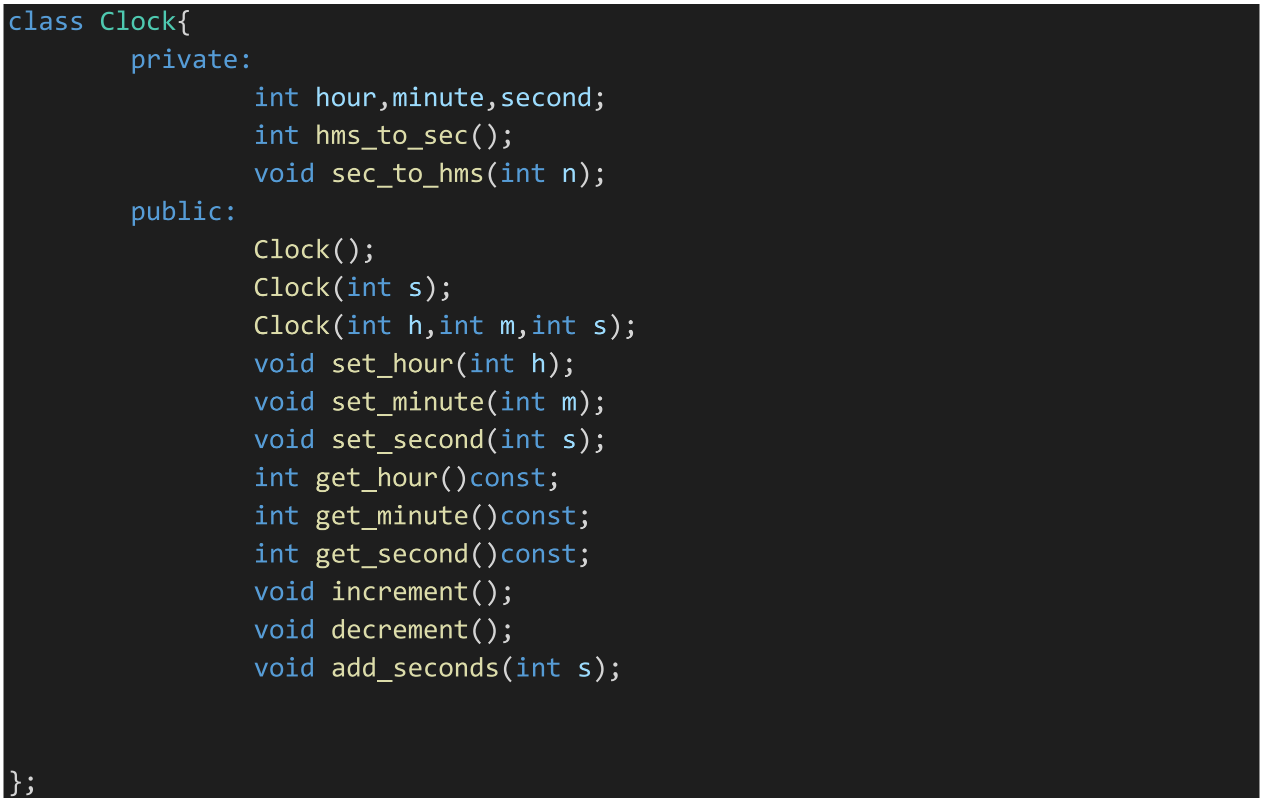
Exercise A


Point 1
Stack
AR
Cplx : Cplx
real 34
imag 5
this
no args
AR
main
num2
realM 34
imagM 5
num1
realM 666
imagM -99
NO ARGS


Point 2
Stack
AR
cplx::setImaginaryPart
this
imagM 500
arg 500
AR
Main
num1
realM 666
imagM 500
num2
realM 34
imagM 5
no args

Exercise C

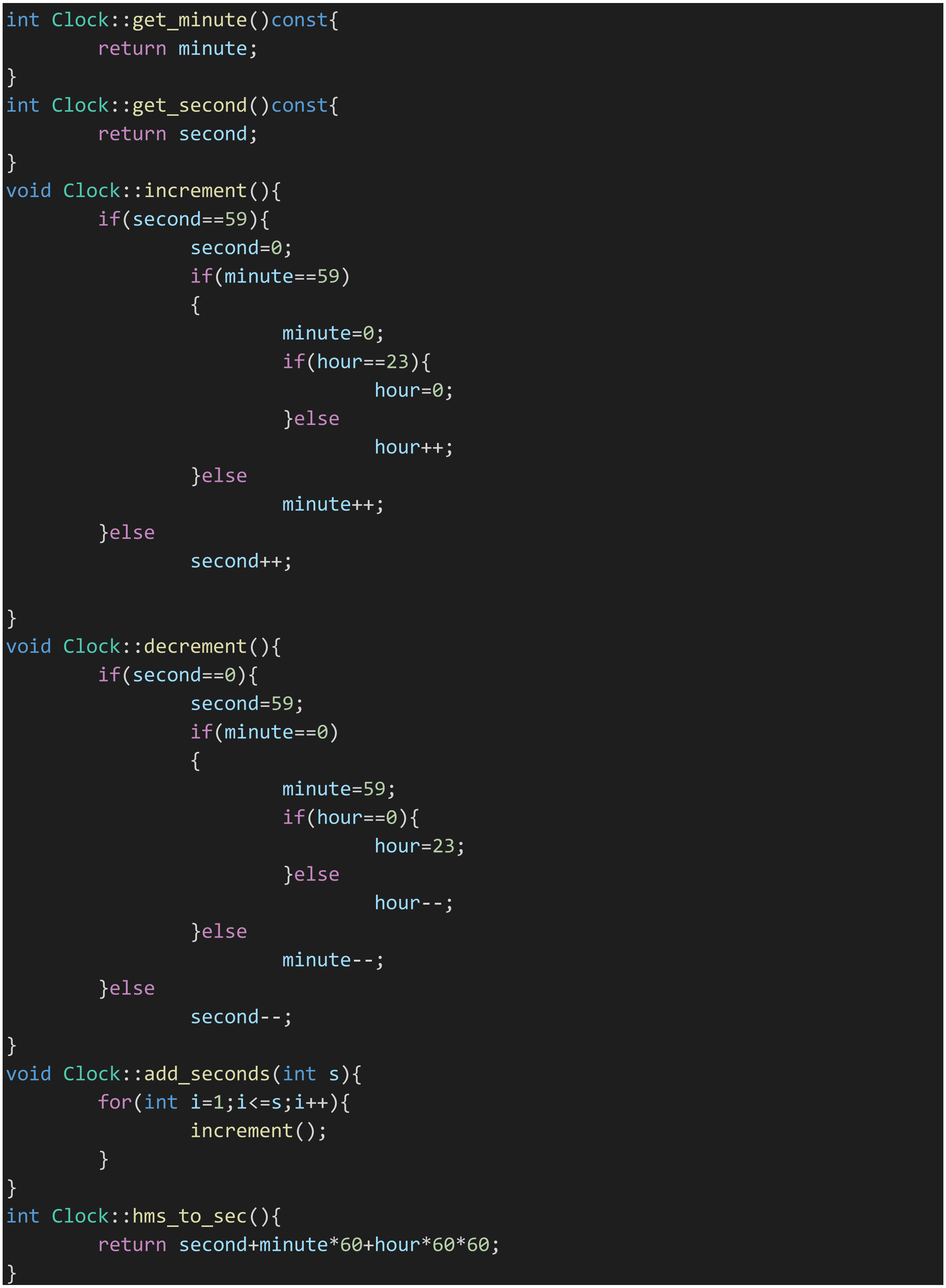
clock.h



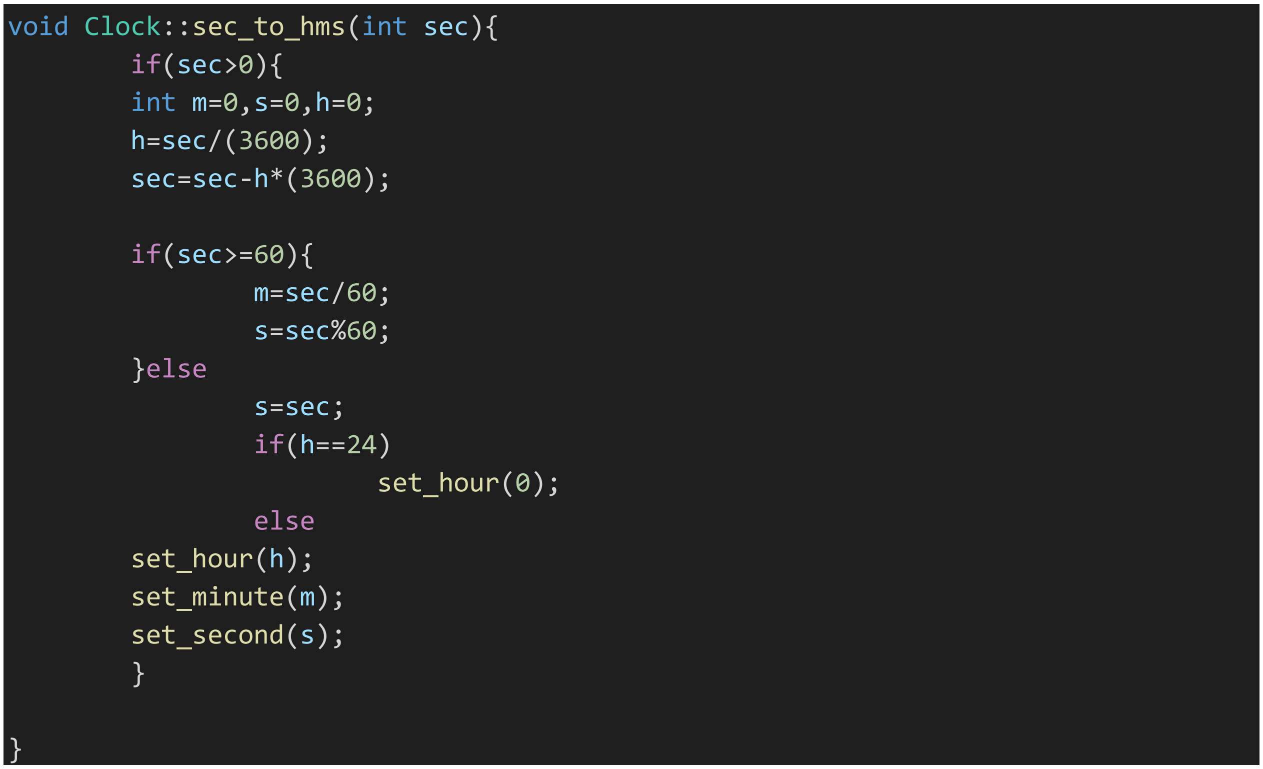
```
class Clock{
        private:
                int hour,minute,second;
                int hms_to_sec();
                void sec_to_hms(int n);
        public:
                Clock();
                Clock(int s);
                Clock(int h,int m,int s);
                void set_hour(int h);
                void set_minute(int m);
                void set_second(int s);
                int get_hour()const;
                int get_minute()const;
                int get_second()const;
                void increment();
                void decrement();
                void add_seconds(int s);


};
```

clock.cpp

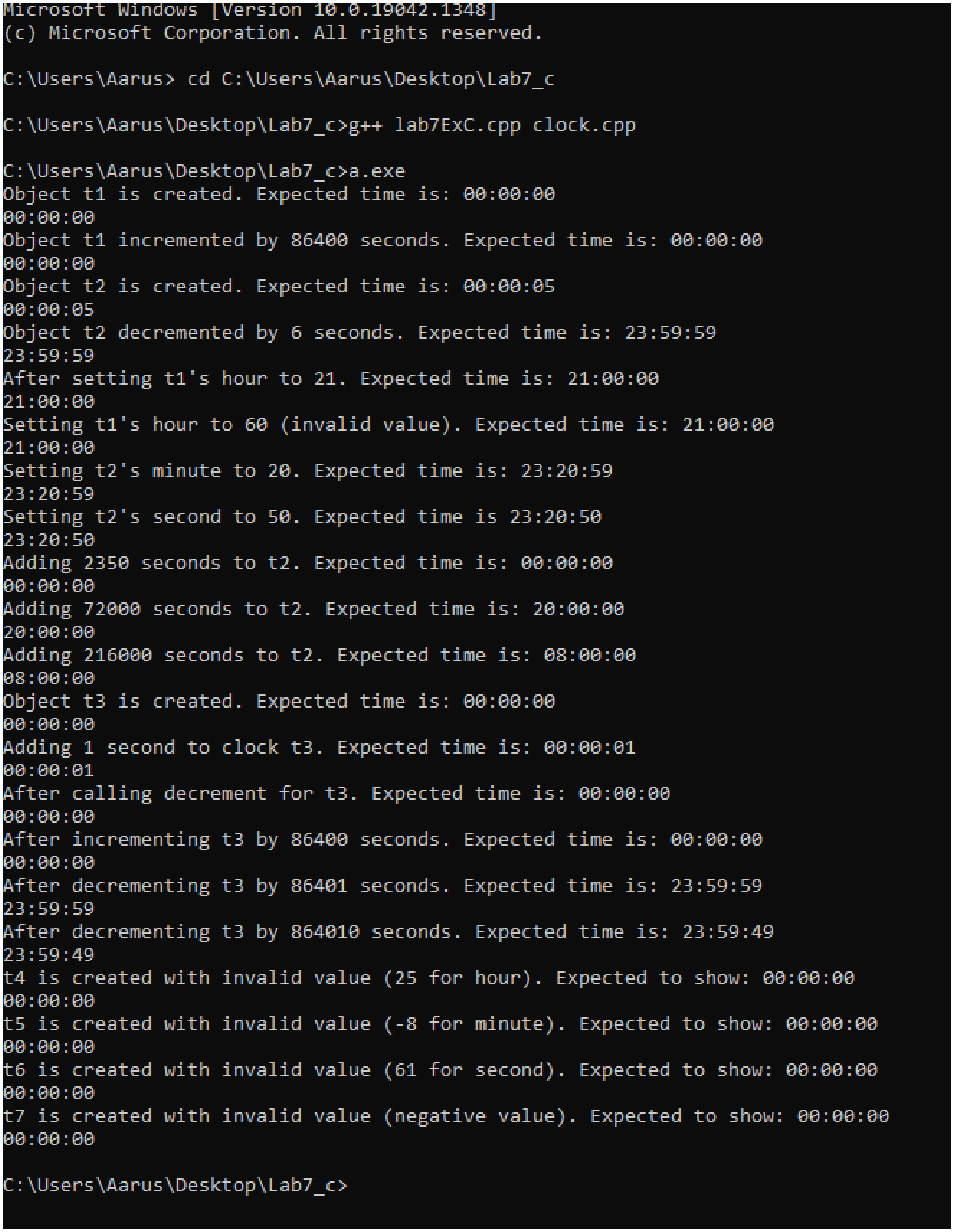
```
#include "clock.h"
Clock::Clock(){
        hour=0;
        minute=0;
        second=0;
}
Clock::Clock(int s){
                hour=0;
        minute=0;
        second=0;
        sec_to_hms(s);
}
Clock::Clock(int h,int m,int s){
        hour=0;
        minute=0;
        second=0;
        if(h>=0&&m<=23&&m>=0&&m<=59&&s>=0&&s<=59){
                        set_hour(h);

                set_minute(m);
                set_second(s);

        }
}
void Clock::set_hour(int h){
        if(h>=0&&h<=23){
                hour=h;
        }
}
void Clock::set_minute(int m){
        if(m>=0&&m<=59){
                minute=m;
        }
}
void Clock::set_second(int s){
        if(s>=0&&s<=59){
                second=s;
        }
}
int Clock::get_hour()const{
        return hour;
}
```



```
int Clock::get_minute()const{
        return minute;
}
int Clock::get_second()const{
        return second;
}
void Clock::increment(){
        if(second==59){
                second=0;
                if(minute==59)
                {
                        minute=0;
                        if(hour==23){
                                hour=0;
                        }else
                                hour++;
                }else
                        minute++;
        }else
                second++;

}
void Clock::decrement(){
        if(second==0){
                second=59;
                if(minute==0)
                {
                        minute=59;
                        if(hour==0){
                                hour=23;
                        }else
                                hour--;
                }else
                        minute--;
        }else
                second--;
}
void Clock::add_seconds(int s){
        for(int i=1;i<=s;i++){
                increment();
        }
}
int Clock::hms_to_sec(){
        return second+minute*60+hour*60*60;
}
```

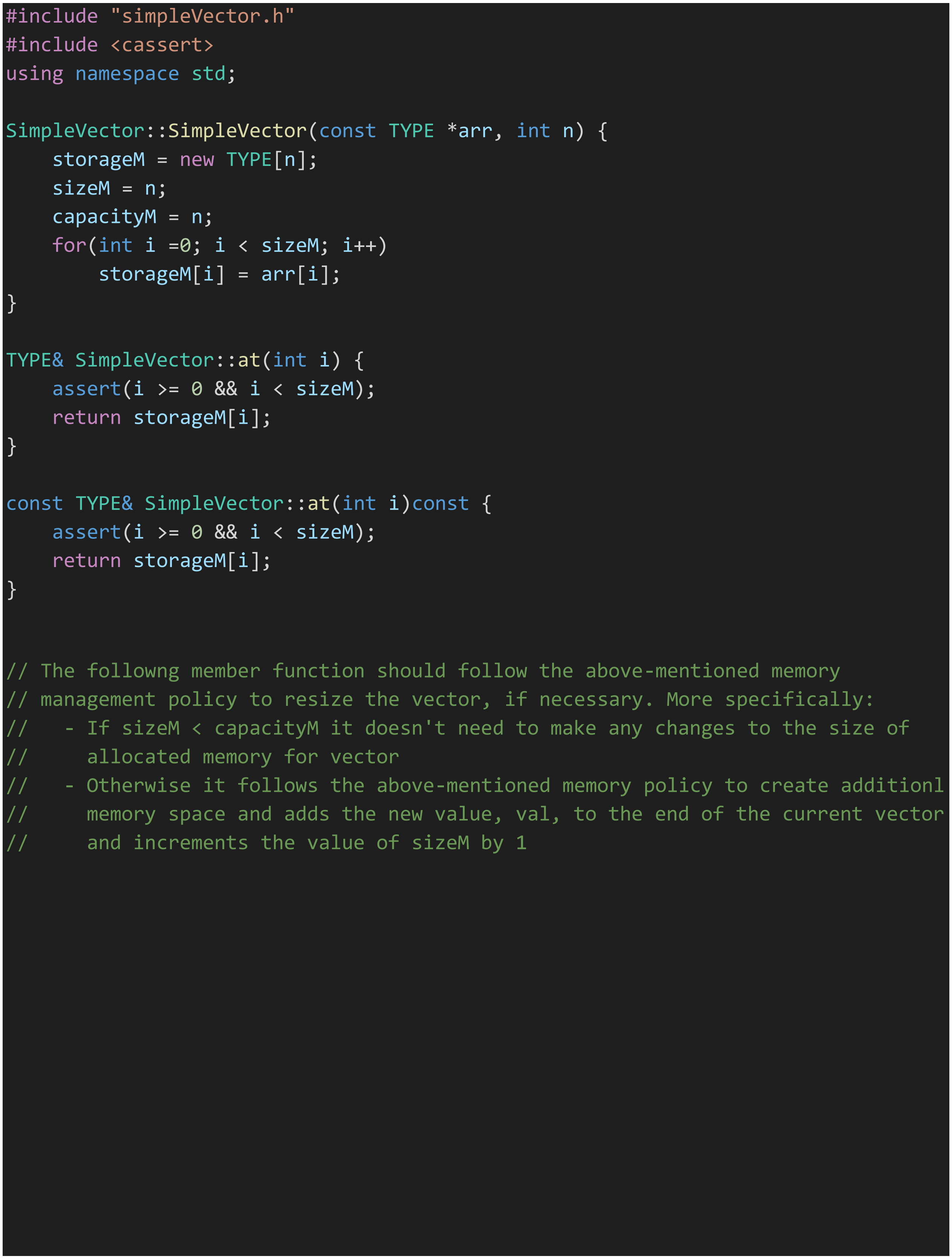


```
void Clock::sec_to_hms(int sec){
        if(sec>0){
        int m=0,s=0,h=0;
        h=sec/(3600);
        sec=sec-h*(3600);

        if(sec>=60){
                m=sec/60;
                s=sec%60;
        }else
                s=sec;
                if(h==24)
                        set_hour(0);
                else
        set_hour(h);
        set_minute(m);
        set_second(s);
        }

}
```

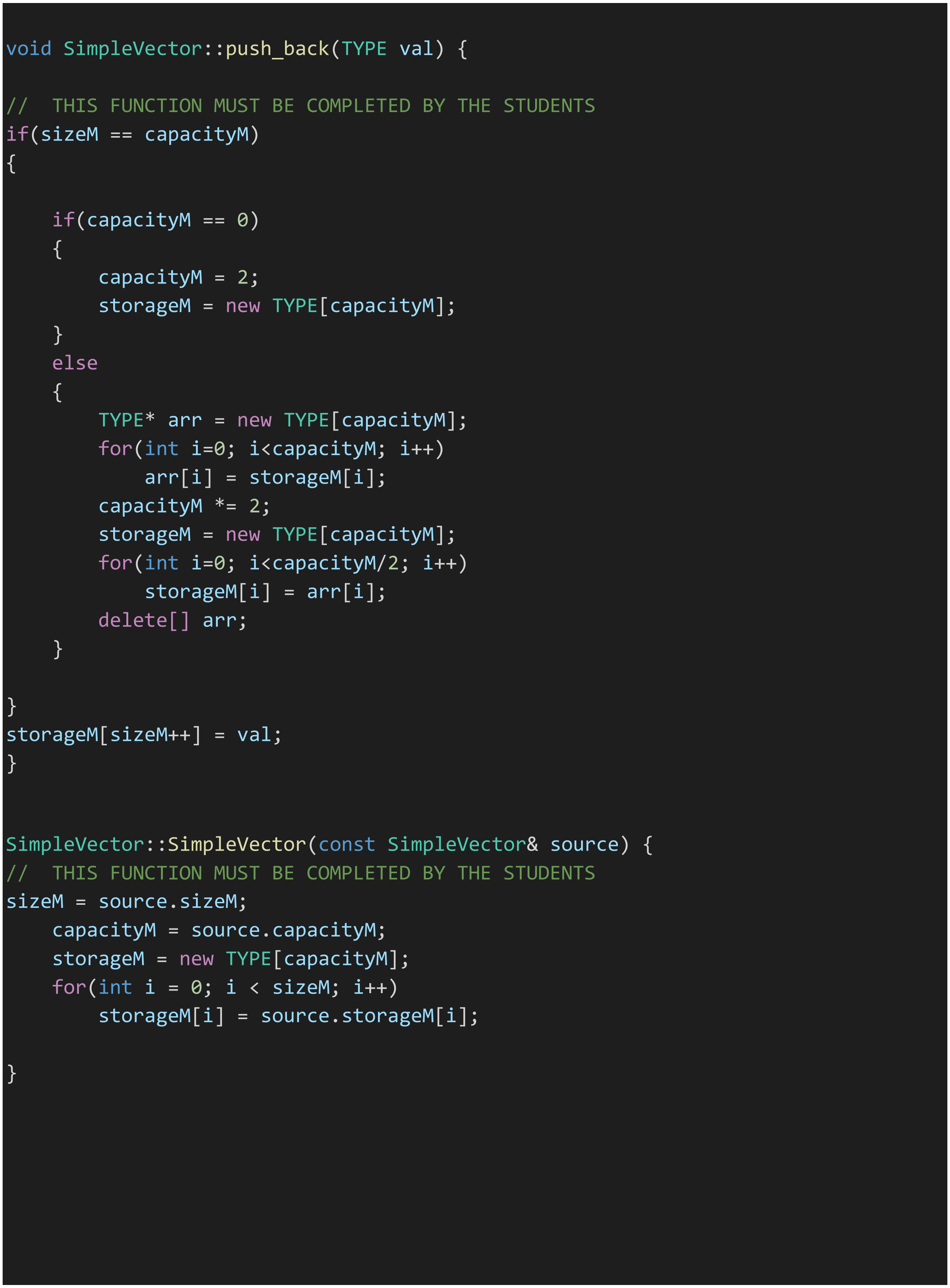


```
Microsoft Windows [Version 10.0.19042.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Aarus> cd C:\Users\Aarus\Desktop\Lab7_c

C:\Users\Aarus\Desktop\Lab7_c>g++ lab7ExC.cpp clock.cpp

C:\Users\Aarus\Desktop\Lab7_c>a.exe
Object t1 is created. Expected time is: 00:00:00
00:00:00
Object t1 incremented by 86400 seconds. Expected time is: 00:00:00
00:00:00
Object t2 is created. Expected time is: 00:00:05
00:00:05
Object t2 decremented by 6 seconds. Expected time is: 23:59:59
23:59:59
After setting t1's hour to 21. Expected time is: 21:00:00
21:00:00
Setting t1's hour to 60 (invalid value). Expected time is: 21:00:00
21:00:00
Setting t2's minute to 20. Expected time is: 23:20:59
23:20:59
Setting t2's second to 50. Expected time is 23:20:50
23:20:50
Adding 2350 seconds to t2. Expected time is: 00:00:00
00:00:00
Adding 72000 seconds to t2. Expected time is: 20:00:00
20:00:00
Adding 216000 seconds to t2. Expected time is: 08:00:00
08:00:00
Object t3 is created. Expected time is: 00:00:00
00:00:00
Adding 1 second to clock t3. Expected time is: 00:00:01
00:00:01
After calling decrement for t3. Expected time is: 00:00:00
00:00:00
After incrementing t3 by 86400 seconds. Expected time is: 00:00:00
00:00:00
After decrementing t3 by 86401 seconds. Expected time is: 23:59:59
23:59:59
After decrementing t3 by 864010 seconds. Expected time is: 23:59:49
23:59:49
t4 is created with invalid value (25 for hour). Expected to show: 00:00:00
00:00:00
t5 is created with invalid value (-8 for minute). Expected to show: 00:00:00
00:00:00
t6 is created with invalid value (61 for second). Expected to show: 00:00:00
00:00:00
t7 is created with invalid value (negative value). Expected to show: 00:00:00
00:00:00

C:\Users\Aarus\Desktop\Lab7_c>
```

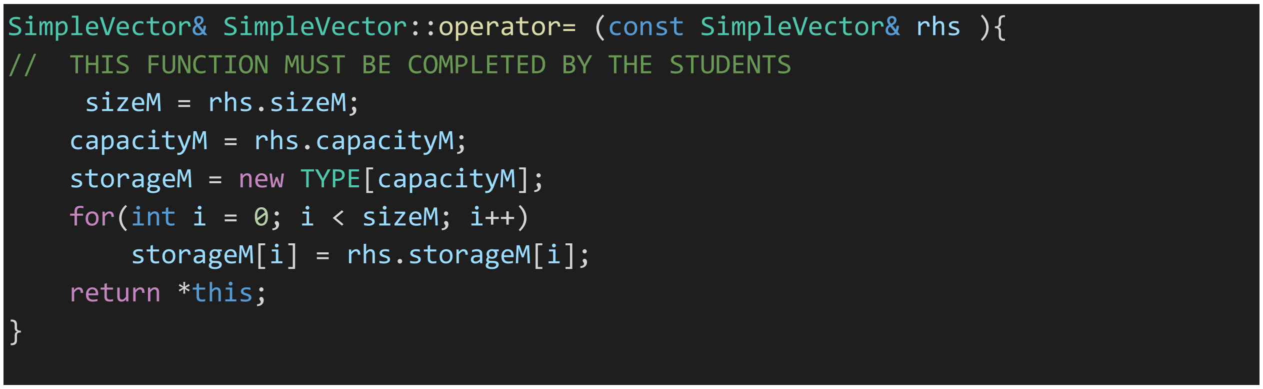
Exercise D



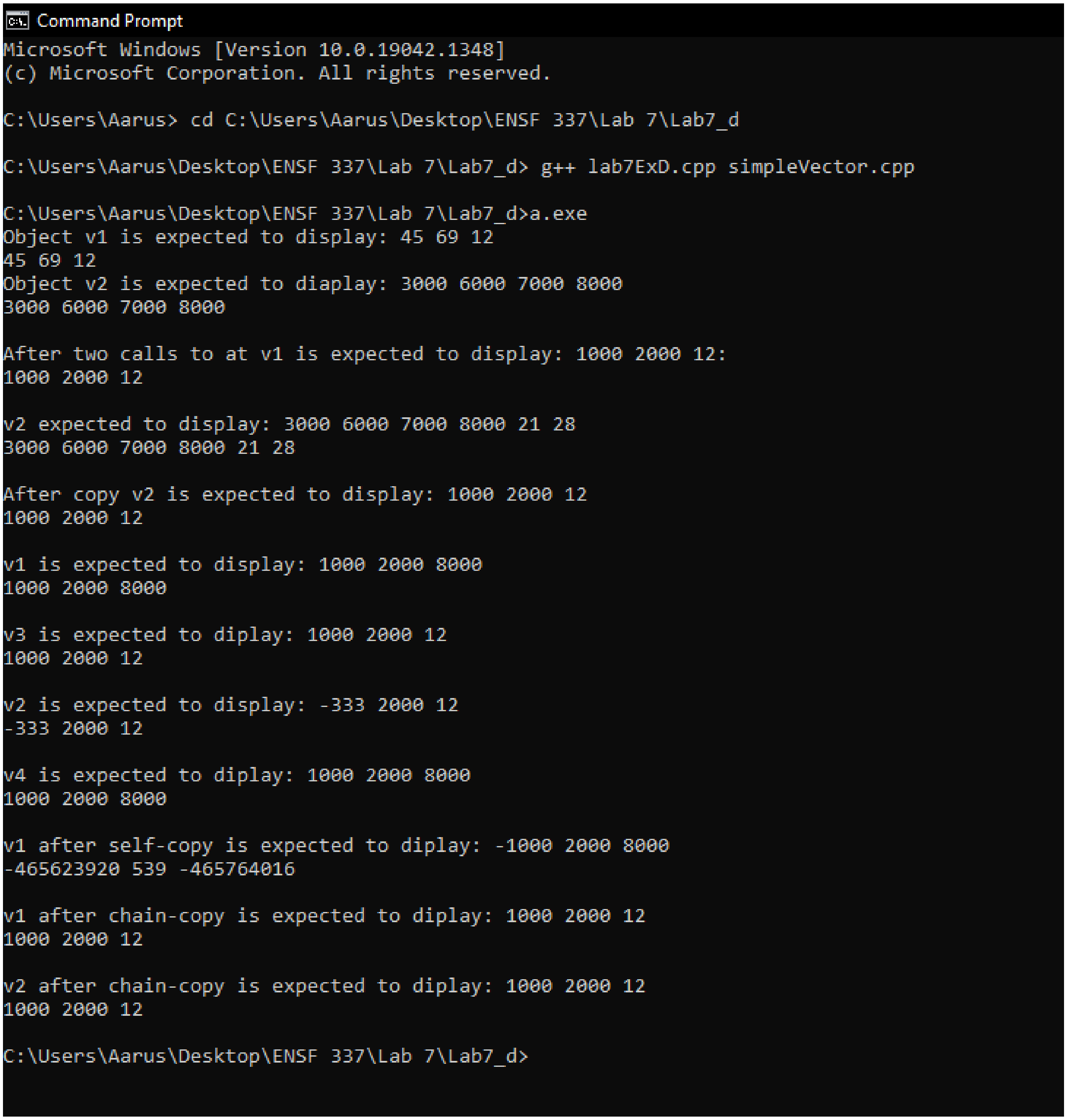
```cpp
#include "simpleVector.h"
#include <cassert>
using namespace std;

SimpleVector::SimpleVector(const TYPE *arr, int n) {
    storageM = new TYPE[n];
    sizeM = n;
    capacityM = n;
    for(int i =0; i < sizeM; i++)
        storageM[i] = arr[i];
}

TYPE& SimpleVector::at(int i) {
    assert(i >= 0 && i < sizeM);
    return storageM[i];
}

const TYPE& SimpleVector::at(int i)const {
    assert(i >= 0 && i < sizeM);
    return storageM[i];
}


// The followng member function should follow the above-mentioned memory
// management policy to resize the vector, if necessary. More specifically:
//   - If sizeM < capacityM it doesn't need to make any changes to the size of
//     allocated memory for vector
//   - Otherwise it follows the above-mentioned memory policy to create additionl
//     memory space and adds the new value, val, to the end of the current vector
//     and increments the value of sizeM by 1
```



```
void SimpleVector::push_back(TYPE val) {

//  THIS FUNCTION MUST BE COMPLETED BY THE STUDENTS
if(sizeM == capacityM)
{

    if(capacityM == 0)
    {
        capacityM = 2;
        storageM = new TYPE[capacityM];
    }
    else
    {
        TYPE* arr = new TYPE[capacityM];
        for(int i=0; i<capacityM; i++)
            arr[i] = storageM[i];
        capacityM *= 2;
        storageM = new TYPE[capacityM];
        for(int i=0; i<capacityM/2; i++)
            storageM[i] = arr[i];
        delete[] arr;
    }

}
storageM[sizeM++] = val;
}


SimpleVector::SimpleVector(const SimpleVector& source) {
//  THIS FUNCTION MUST BE COMPLETED BY THE STUDENTS
sizeM = source.sizeM;
    capacityM = source.capacityM;
    storageM = new TYPE[capacityM];
    for(int i = 0; i < sizeM; i++)
        storageM[i] = source.storageM[i];

}
```

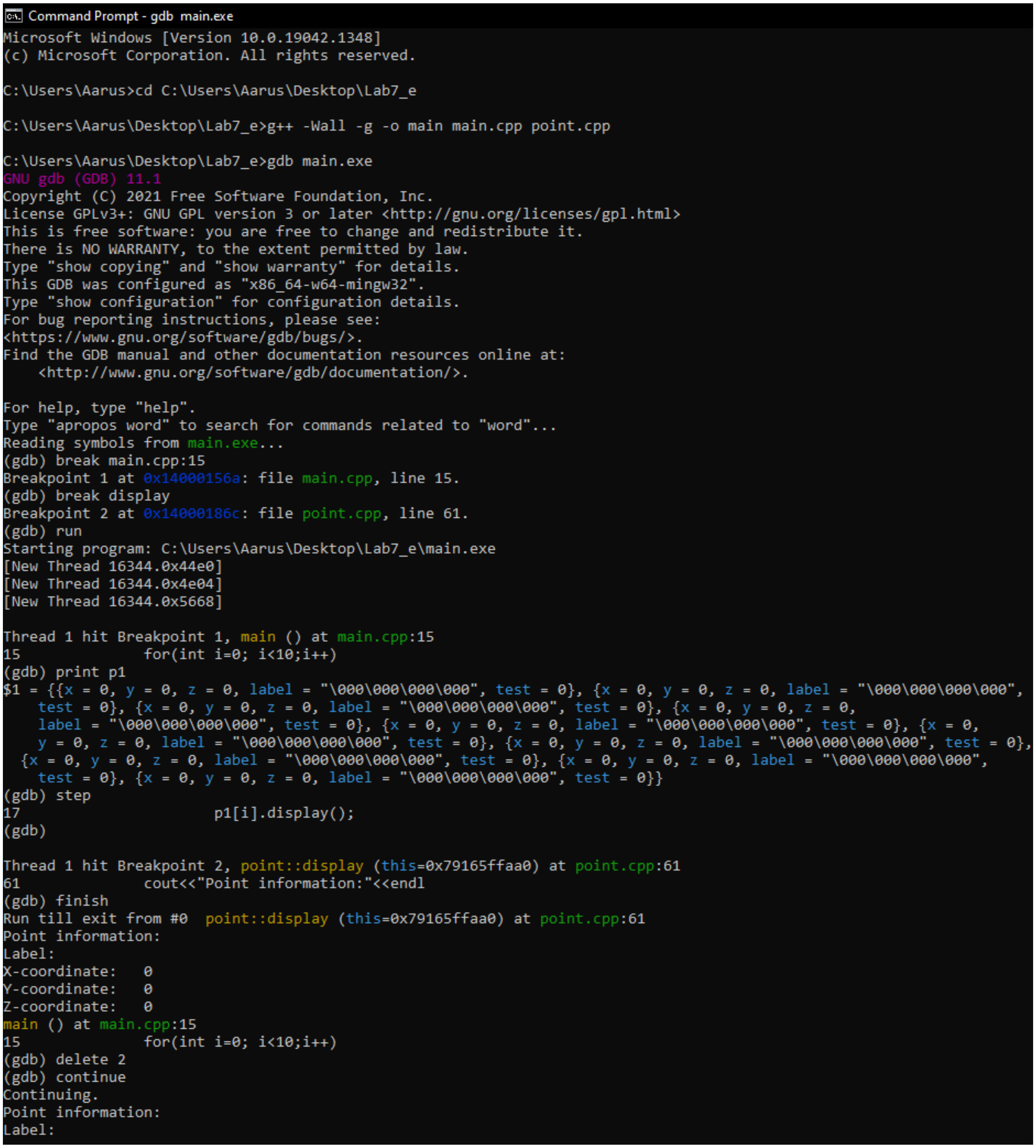


```
SimpleVector& SimpleVector::operator= (const SimpleVector& rhs ){
//  THIS FUNCTION MUST BE COMPLETED BY THE STUDENTS
     sizeM = rhs.sizeM;
    capacityM = rhs.capacityM;
    storageM = new TYPE[capacityM];
    for(int i = 0; i < sizeM; i++)
        storageM[i] = rhs.storageM[i];
    return *this;
}
```

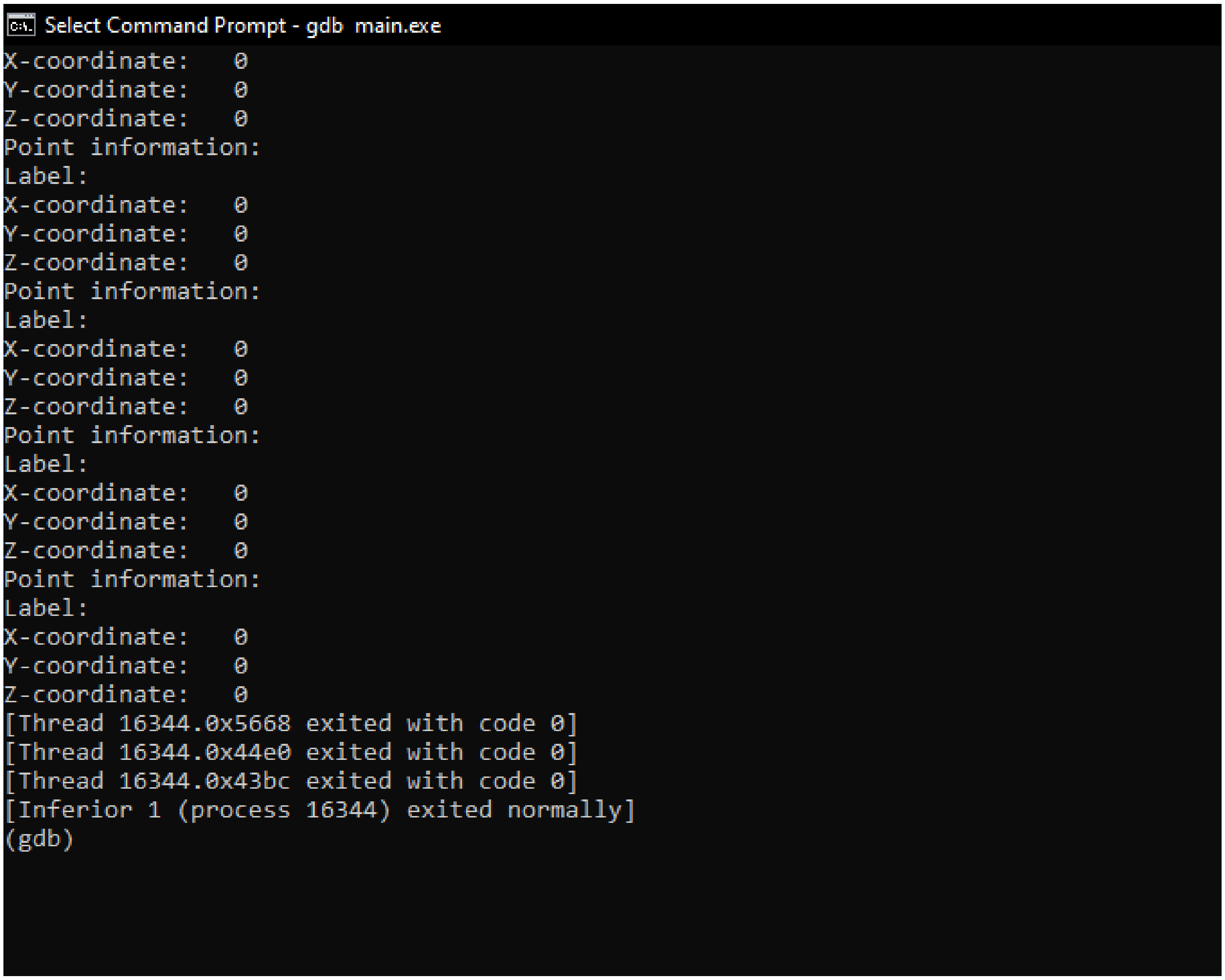


```
Microsoft Windows [Version 10.0.19042.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Aarus> cd C:\Users\Aarus\Desktop\ENSF 337\Lab 7\Lab7_d

C:\Users\Aarus\Desktop\ENSF 337\Lab 7\Lab7_d> g++ lab7ExD.cpp simpleVector.cpp

C:\Users\Aarus\Desktop\ENSF 337\Lab 7\Lab7_d>a.exe
Object v1 is expected to display: 45 69 12
45 69 12
Object v2 is expected to diaplay: 3000 6000 7000 8000
3000 6000 7000 8000

After two calls to at v1 is expected to display: 1000 2000 12:
1000 2000 12

v2 expected to display: 3000 6000 7000 8000 21 28
3000 6000 7000 8000 21 28

After copy v2 is expected to display: 1000 2000 12
1000 2000 12

v1 is expected to display: 1000 2000 8000
1000 2000 8000

v3 is expected to diplay: 1000 2000 12
1000 2000 12

v2 is expected to display: -333 2000 12
-333 2000 12

v4 is expected to diplay: 1000 2000 8000
1000 2000 8000

v1 after self-copy is expected to diplay: -1000 2000 8000
-465623920 539 -465764016

v1 after chain-copy is expected to diplay: 1000 2000 12
1000 2000 12

v2 after chain-copy is expected to diplay: 1000 2000 12
1000 2000 12

C:\Users\Aarus\Desktop\ENSF 337\Lab 7\Lab7_d>
```

Exercise E



```
Microsoft Windows [Version 10.0.19042.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Aarus>cd C:\Users\Aarus\Desktop\Lab7_e

C:\Users\Aarus\Desktop\Lab7_e>g++ -Wall -g -o main main.cpp point.cpp

C:\Users\Aarus\Desktop\Lab7_e>gdb main.exe
GNU gdb (GDB) 11.1
Copyright (C) 2021 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:
<https://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 main.exe...
(gdb) break main.cpp:15
Breakpoint 1 at 0x14000156a: file main.cpp, line 15.
(gdb) break display
Breakpoint 2 at 0x14000186c: file point.cpp, line 61.
(gdb) run
Starting program: C:\Users\Aarus\Desktop\Lab7_e\main.exe
[New Thread 16344.0x44e0]
[New Thread 16344.0x4e04]
[New Thread 16344.0x5668]

Thread 1 hit Breakpoint 1, main () at main.cpp:15
15                for(int i=0; i<10;i++)
(gdb) print p1
$1 = {{x = 0, y = 0, z = 0, label = "\000\000\000\000", test = 0}, {x = 0, y = 0, z = 0, label = "\000\000\000\000",
    test = 0}, {x = 0, y = 0, z = 0, label = "\000\000\000\000", test = 0}, {x = 0, y = 0, z = 0,
    label = "\000\000\000\000", test = 0}, {x = 0, y = 0, z = 0, label = "\000\000\000\000", test = 0}, {x = 0,
    y = 0, z = 0, label = "\000\000\000\000", test = 0}, {x = 0, y = 0, z = 0, label = "\000\000\000\000", test = 0},
  {x = 0, y = 0, z = 0, label = "\000\000\000\000", test = 0}, {x = 0, y = 0, z = 0, label = "\000\000\000\000",
    test = 0}, {x = 0, y = 0, z = 0, label = "\000\000\000\000", test = 0}}
(gdb) step
17                        p1[i].display();
(gdb)

Thread 1 hit Breakpoint 2, point::display (this=0x79165ffaa0) at point.cpp:61
61                cout<<"Point information:"<<endl
(gdb) finish
Run till exit from #0  point::display (this=0x79165ffaa0) at point.cpp:61
Point information:
Label:
X-coordinate:   0
Y-coordinate:   0
Z-coordinate:   0
main () at main.cpp:15
15                for(int i=0; i<10;i++)
(gdb) delete 2
(gdb) continue
Continuing.
Point information:
Label:
```



```
X-coordinate:   0
Y-coordinate:   0
Z-coordinate:   0
Point information:
Label:
X-coordinate:   0
Y-coordinate:   0
Z-coordinate:   0
Point information:
Label:
X-coordinate:   0
Y-coordinate:   0
Z-coordinate:   0
Point information:
Label:
X-coordinate:   0
Y-coordinate:   0
Z-coordinate:   0
Point information:
Label:
X-coordinate:   0
Y-coordinate:   0
Z-coordinate:   0
[Thread 16344.0x5668 exited with code 0]
[Thread 16344.0x44e0 exited with code 0]
[Thread 16344.0x43bc exited with code 0]
[Inferior 1 (process 16344) exited normally]
(gdb)
```