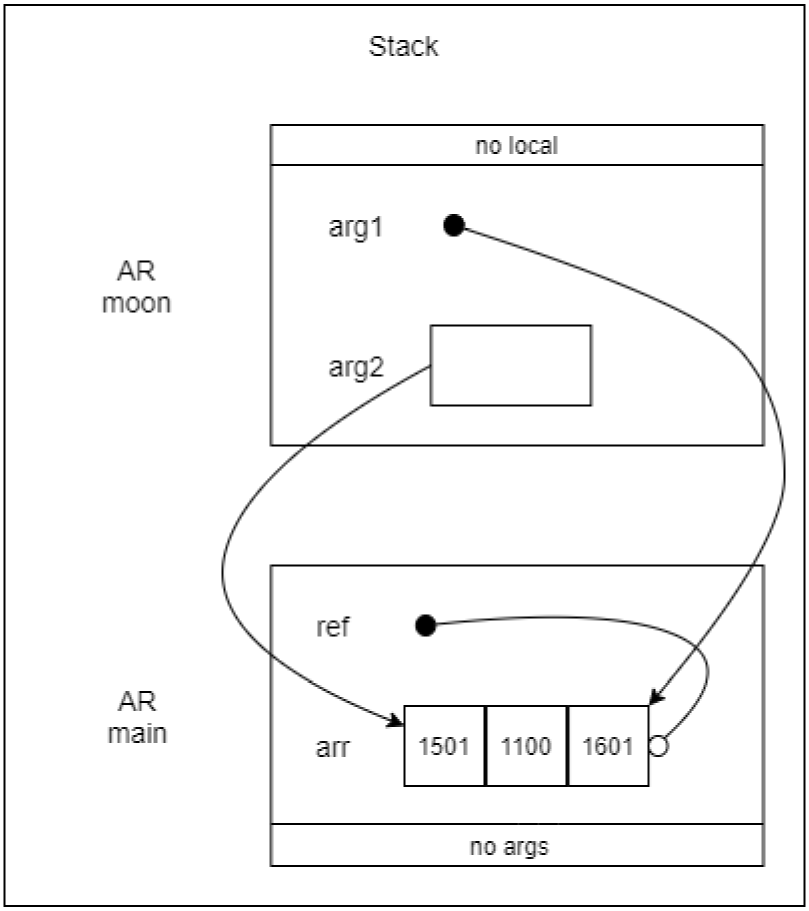
ENSF 614 – Fall 2021

Lab 3 – Tuesday, October 05

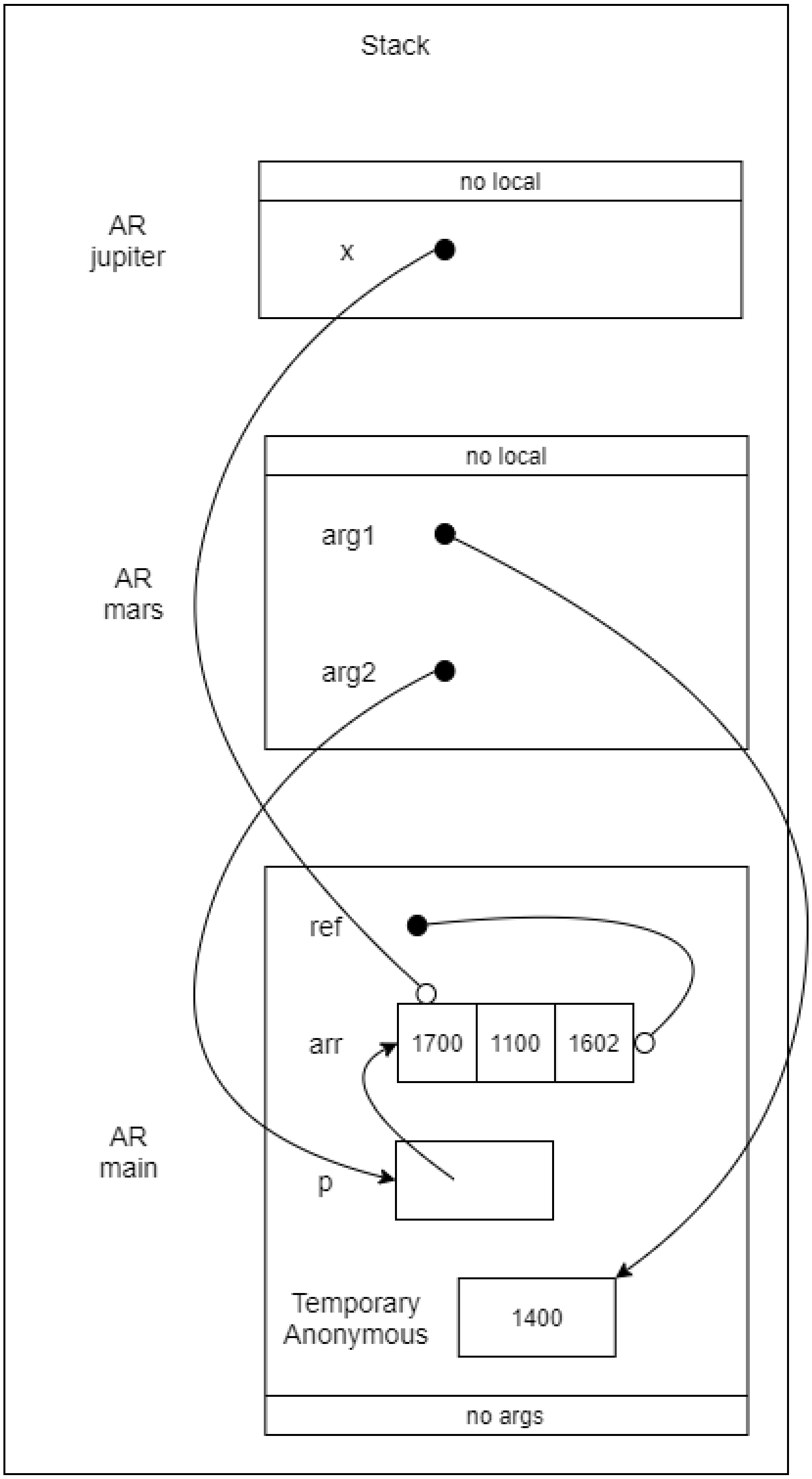
Student Name: Bhavyai Gupta

Submission date: October 05, 2021

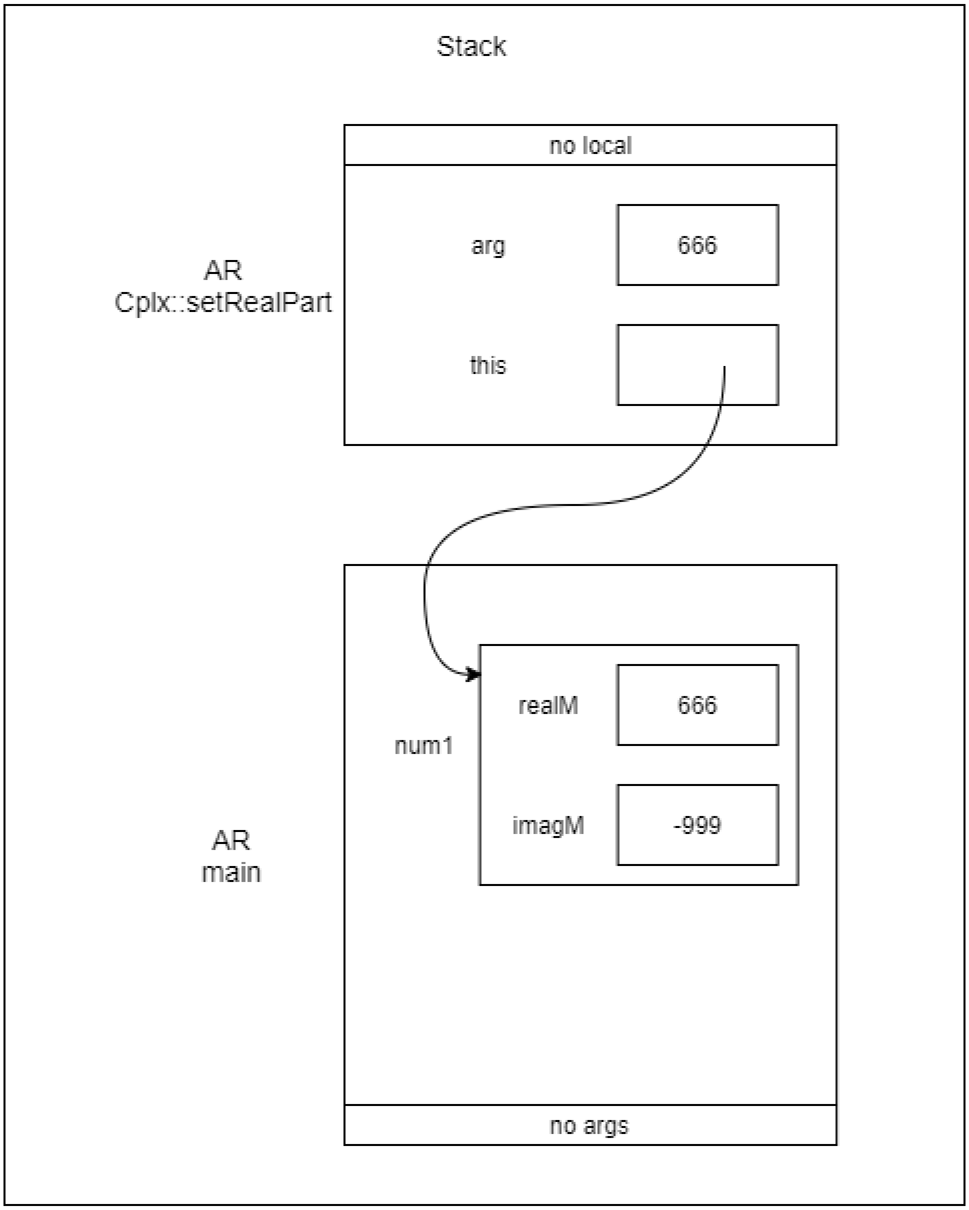
Exercise A – AR Diagram for Point One



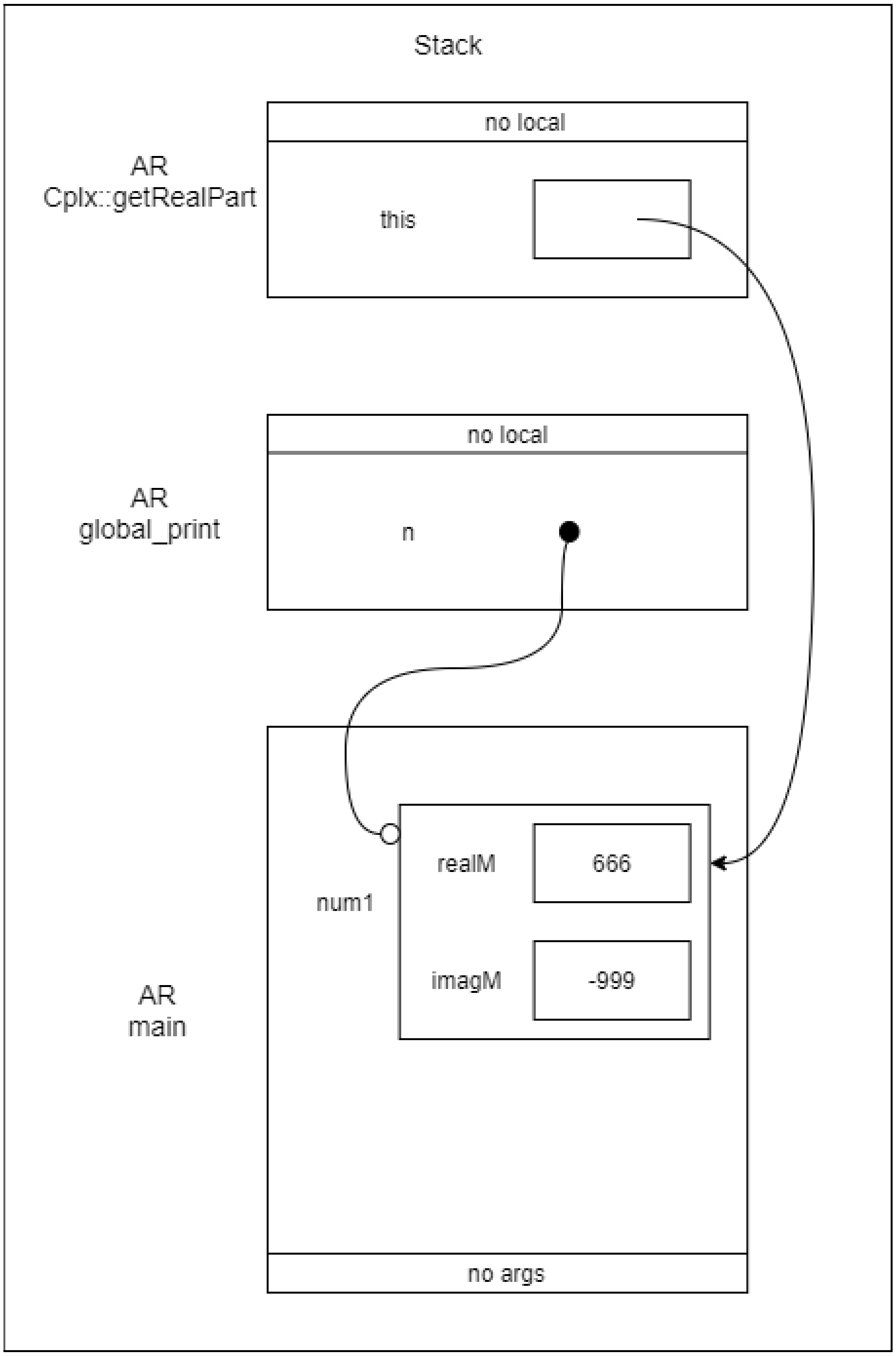
Exercise A – AR Diagram for Point Two



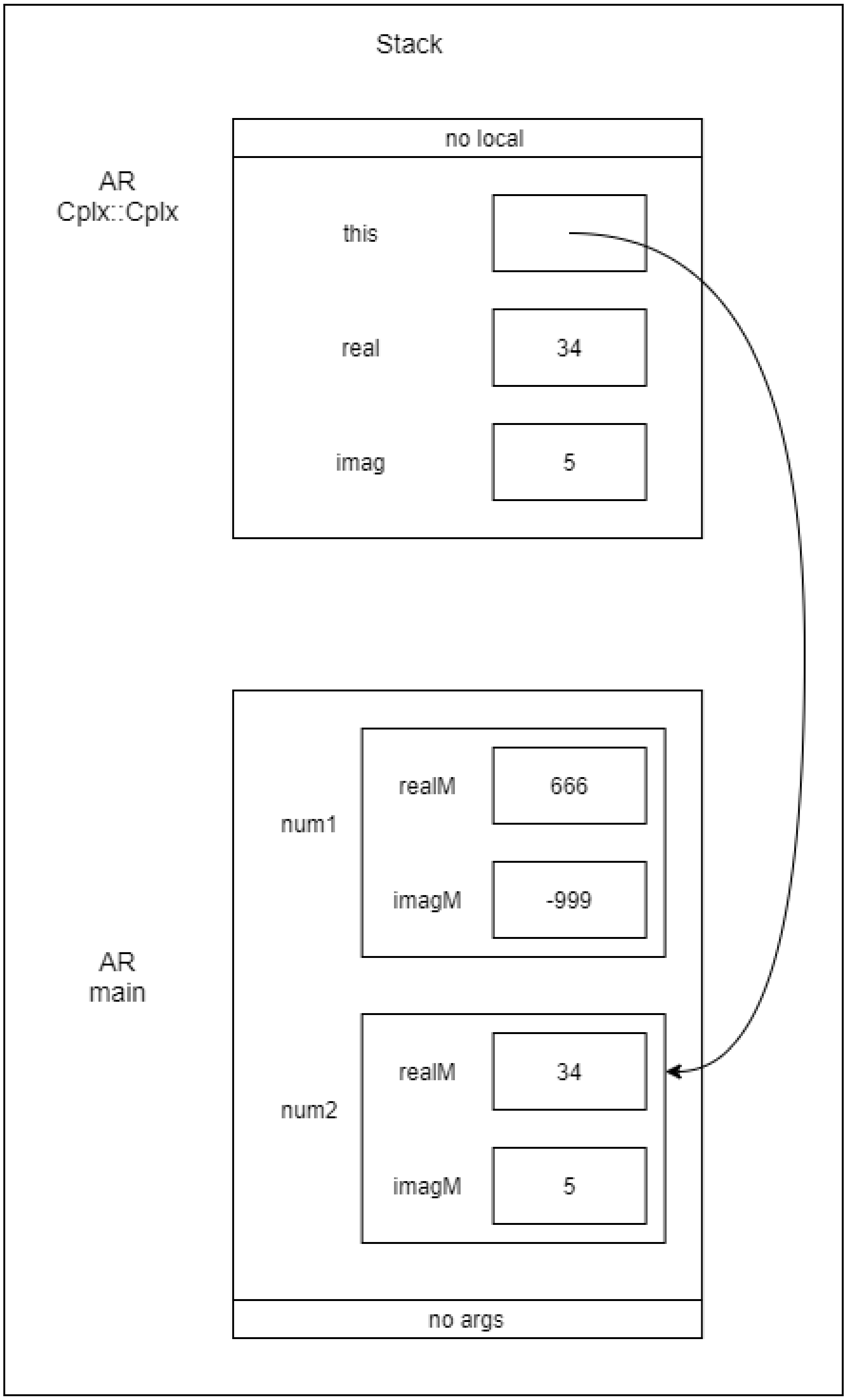
Exercise B – AR Diagram for Point One



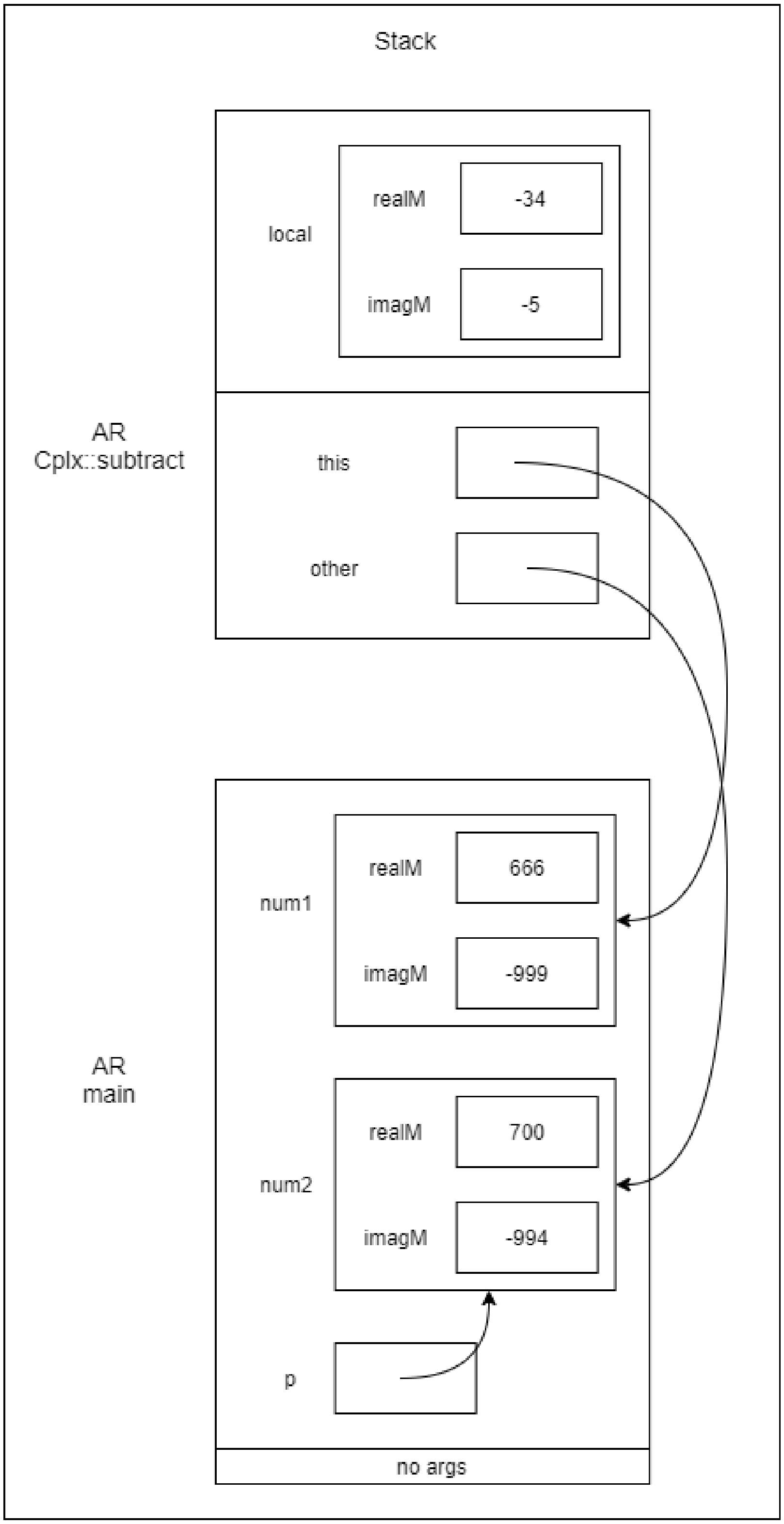
Exercise B – AR Diagram for Point Two



Exercise B – AR Diagram for Point Three

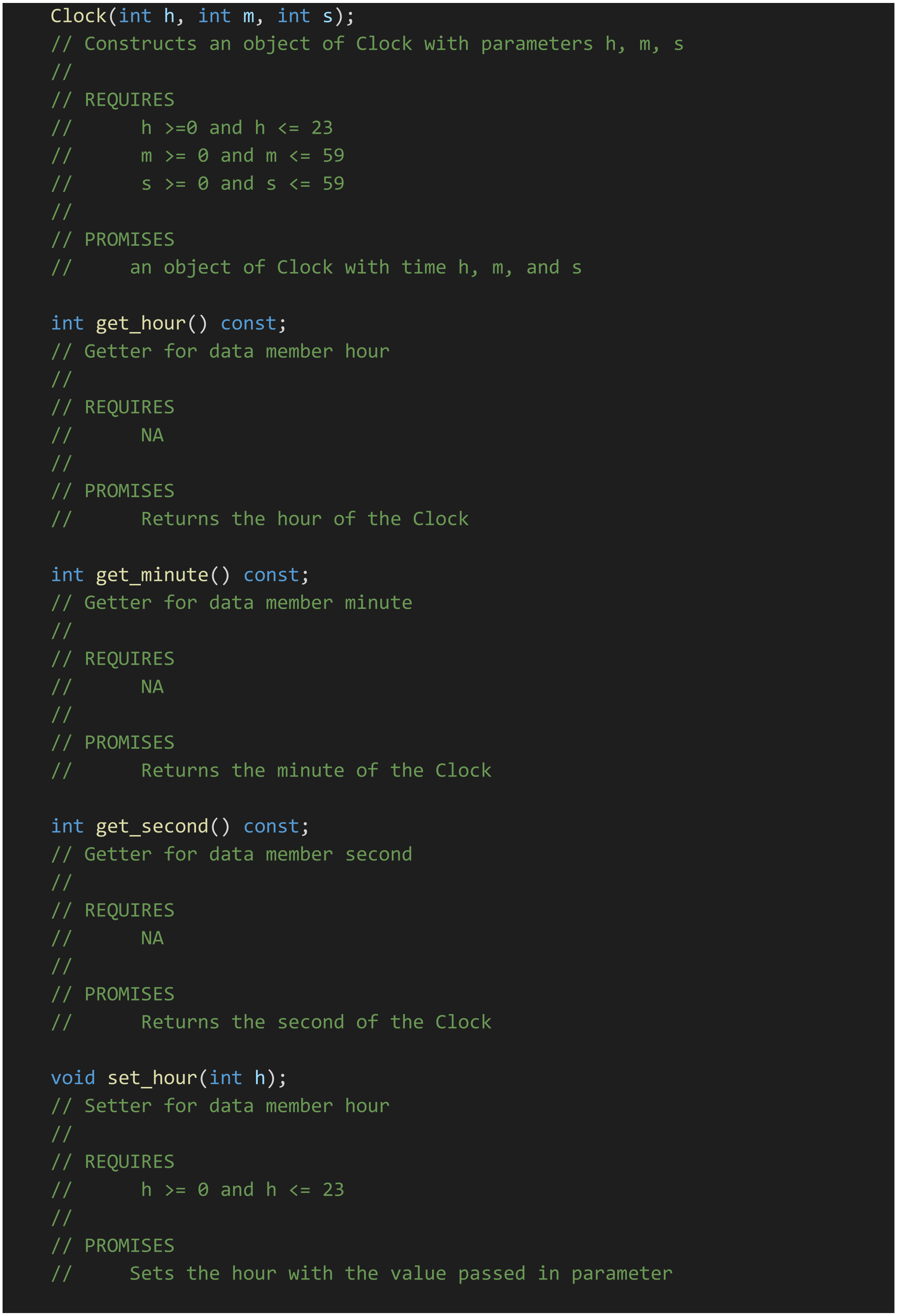


Exercise B – AR Diagram for Point Four

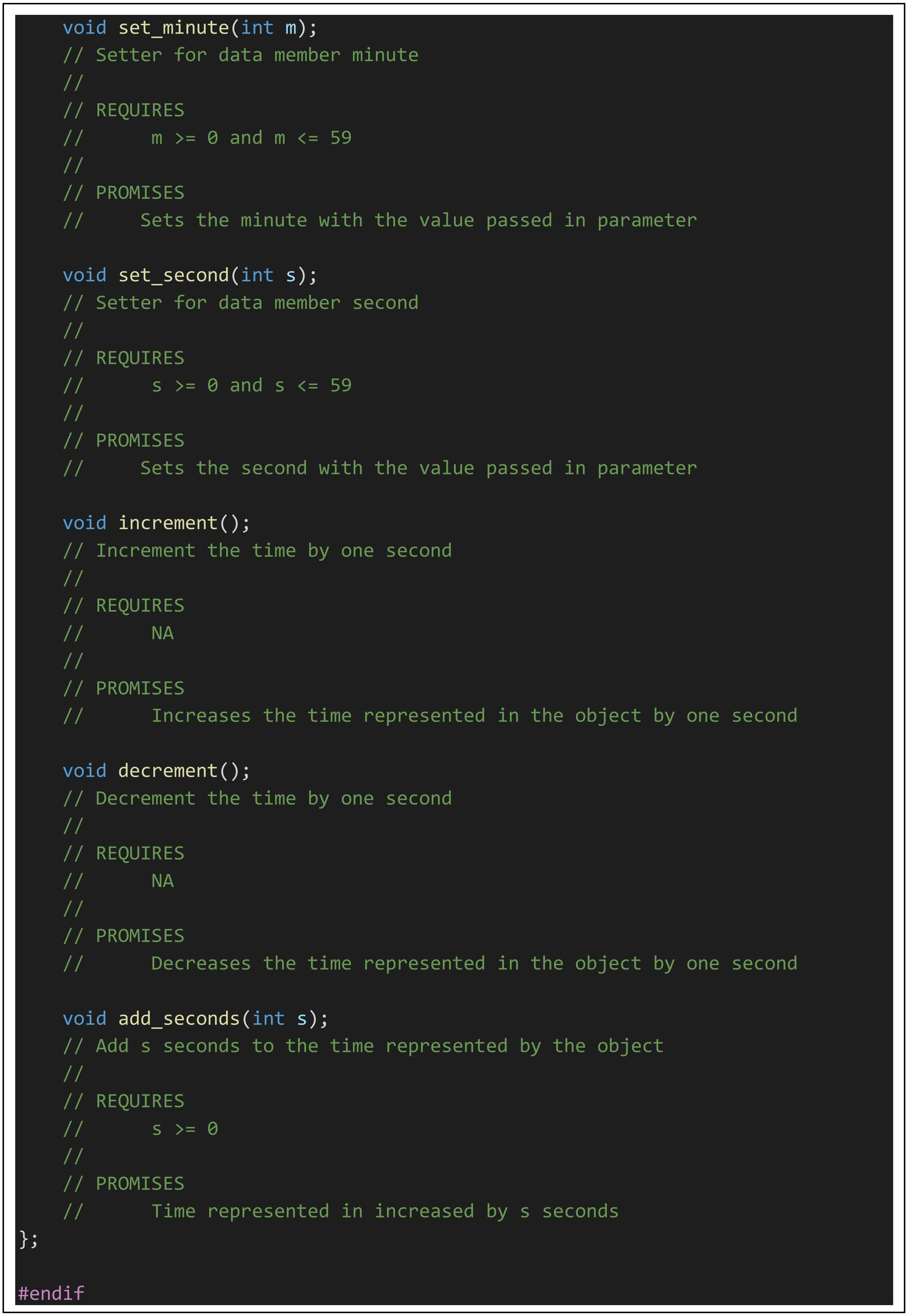


Exercise C – Source File lab3Clock.h

```
/*
 * File Name:               lab3Clock.h
 * Course:                  ENSF 614 - Fall 2021
 * Lab # and Assignment #:  Lab 3 Exercise C
 * Lab section:             B01
 * Completed by:            Bhavyai Gupta
 * Submission Date:         October 05, 2021
 */

#ifndef LAB_3_CLOCK
#define LAB_3_CLOCK

class Clock
{
private:
    int hour;
    int minute;
    int second;

    int hms_to_sec();
    void sec_to_hms(int n);

public:
    Clock();
    // Constructs an object of Clock with all values initialized to 0
    //
    // REQUIRES
    //      NA
    //
    // PROMISES
    //      An object of Clock with all values initialized to 0

    Clock(int s);
    // Constructs an object of Clock with time equivalent to the value
    // of s passed in the argument
    //
    // REQUIRES
    //      s >= 0
    //
    // PROMISES
    //      An object of Clock with time equivalent to the value of s
```

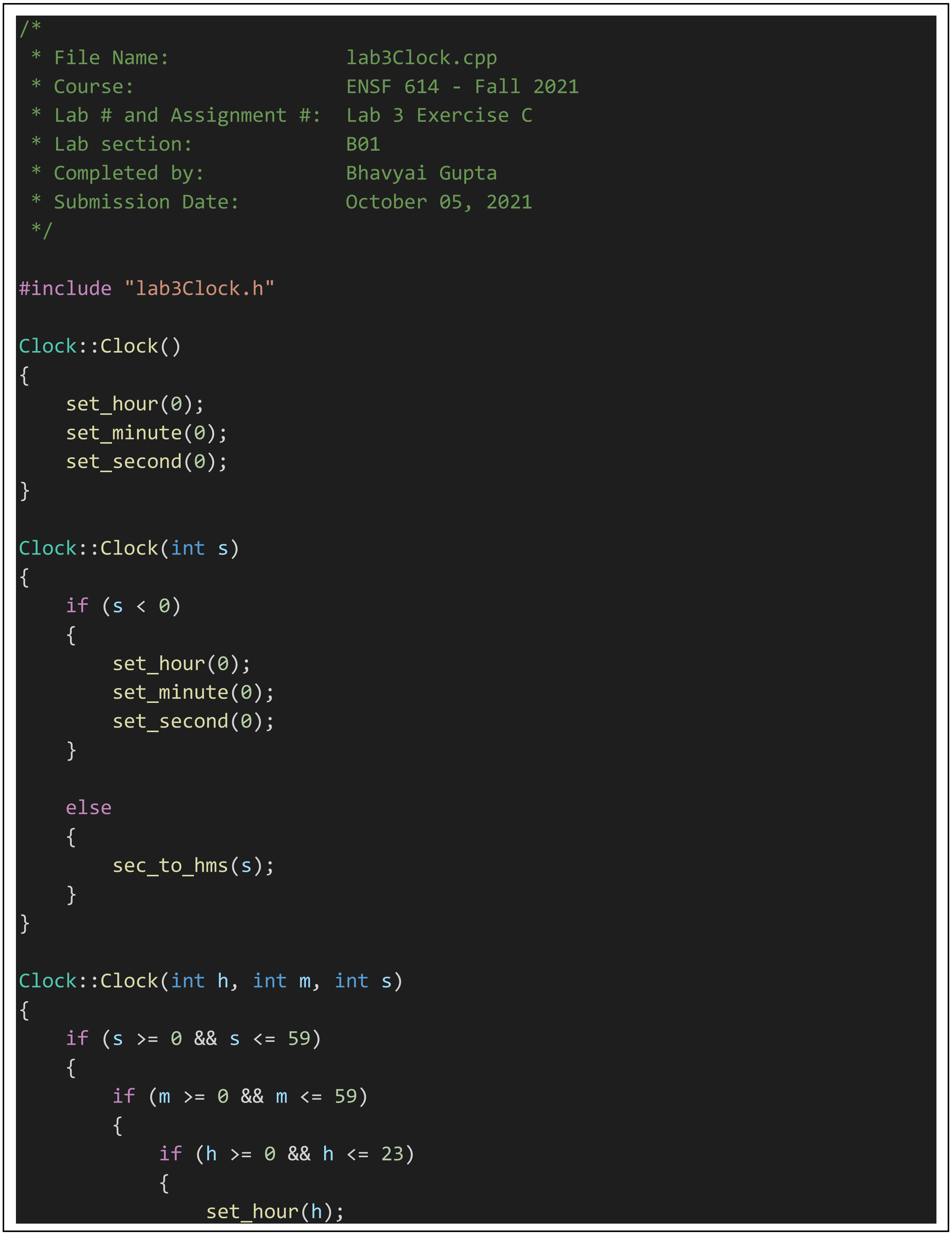


```cpp
    Clock(int h, int m, int s);
    // Constructs an object of Clock with parameters h, m, s
    //
    // REQUIRES
    //      h >=0 and h <= 23
    //      m >= 0 and m <= 59
    //      s >= 0 and s <= 59
    //
    // PROMISES
    //     an object of Clock with time h, m, and s

    int get_hour() const;
    // Getter for data member hour
    //
    // REQUIRES
    //      NA
    //
    // PROMISES
    //      Returns the hour of the Clock

    int get_minute() const;
    // Getter for data member minute
    //
    // REQUIRES
    //      NA
    //
    // PROMISES
    //      Returns the minute of the Clock

    int get_second() const;
    // Getter for data member second
    //
    // REQUIRES
    //      NA
    //
    // PROMISES
    //      Returns the second of the Clock

    void set_hour(int h);
    // Setter for data member hour
    //
    // REQUIRES
    //      h >= 0 and h <= 23
    //
    // PROMISES
    //     Sets the hour with the value passed in parameter
```

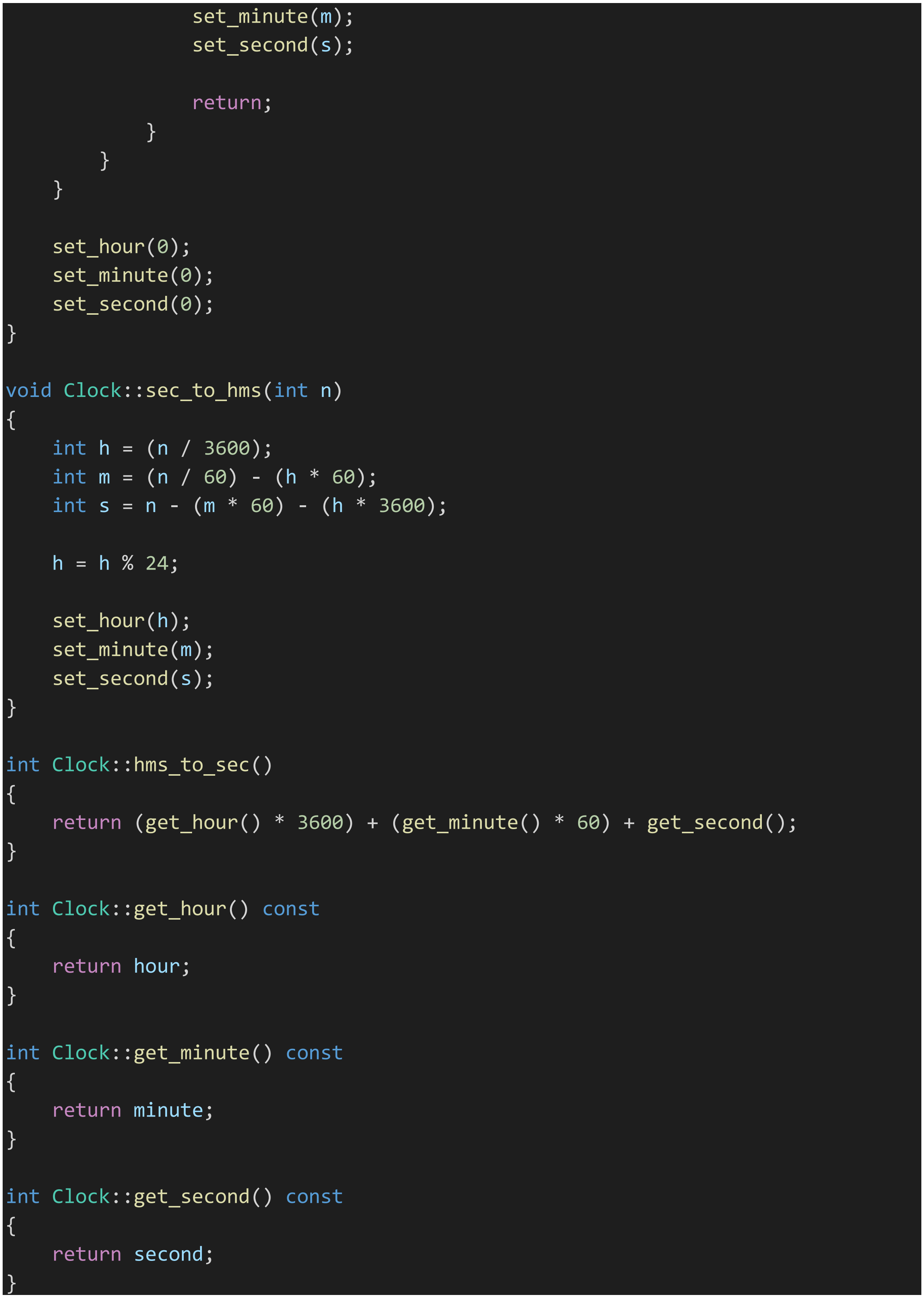


```
    void set_minute(int m);
    // Setter for data member minute
    //
    // REQUIRES
    //      m >= 0 and m <= 59
    //
    // PROMISES
    //     Sets the minute with the value passed in parameter

    void set_second(int s);
    // Setter for data member second
    //
    // REQUIRES
    //      s >= 0 and s <= 59
    //
    // PROMISES
    //     Sets the second with the value passed in parameter

    void increment();
    // Increment the time by one second
    //
    // REQUIRES
    //      NA
    //
    // PROMISES
    //      Increases the time represented in the object by one second

    void decrement();
    // Decrement the time by one second
    //
    // REQUIRES
    //      NA
    //
    // PROMISES
    //      Decreases the time represented in the object by one second

    void add_seconds(int s);
    // Add s seconds to the time represented by the object
    //
    // REQUIRES
    //      s >= 0
    //
    // PROMISES
    //      Time represented in increased by s seconds
};

#endif
```

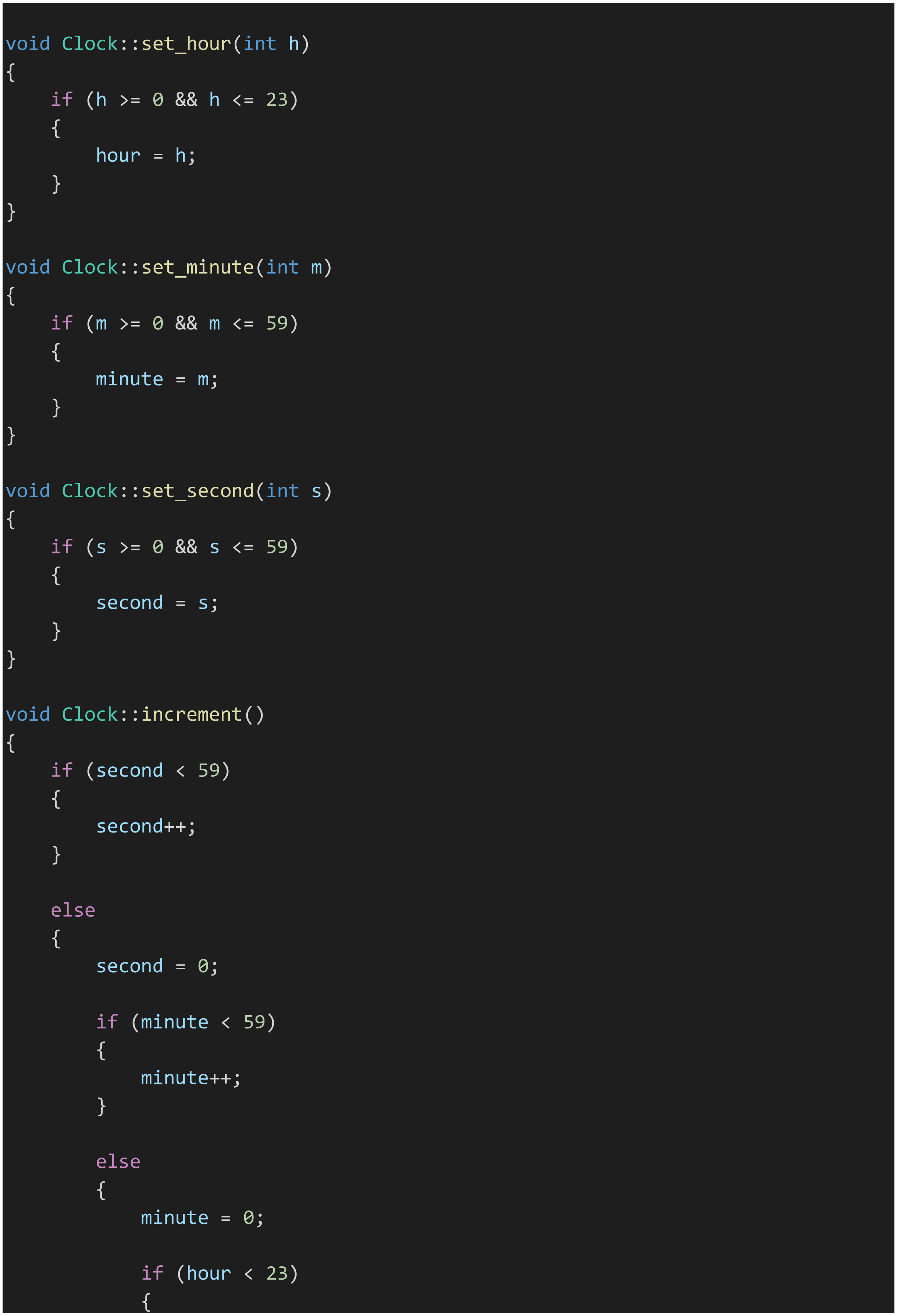
Exercise C – Source File lab3Clock.cpp



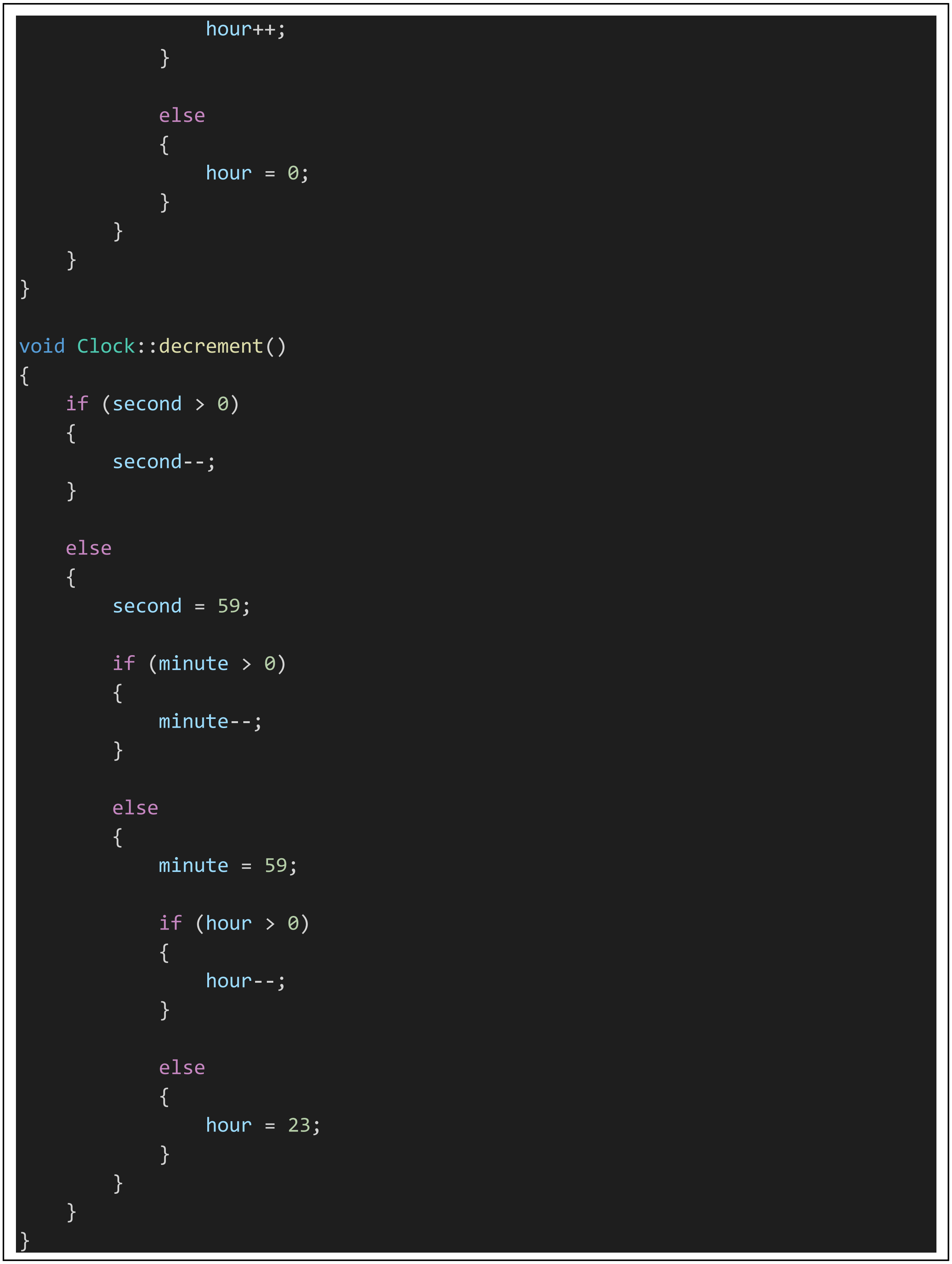
```cpp
/*
 * File Name:                  lab3Clock.cpp
 * Course:                     ENSF 614 - Fall 2021
 * Lab # and Assignment #:     Lab 3 Exercise C
 * Lab section:                B01
 * Completed by:               Bhavyai Gupta
 * Submission Date:            October 05, 2021
 */

#include "lab3Clock.h"

Clock::Clock()
{
    set_hour(0);
    set_minute(0);
    set_second(0);
}

Clock::Clock(int s)
{
    if (s < 0)
    {
        set_hour(0);
        set_minute(0);
        set_second(0);
    }

    else
    {
        sec_to_hms(s);
    }
}

Clock::Clock(int h, int m, int s)
{
    if (s >= 0 && s <= 59)
    {
        if (m >= 0 && m <= 59)
        {
            if (h >= 0 && h <= 23)
            {
                set_hour(h);
```



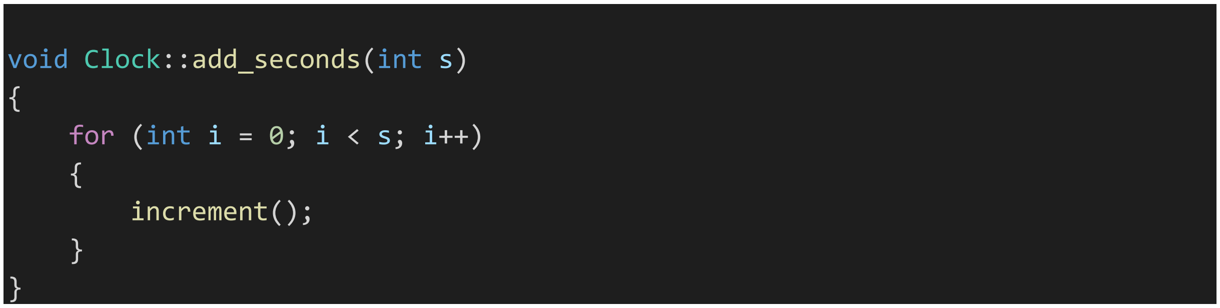
```
                set_minute(m);
                set_second(s);

                return;
            }
        }
    }

    set_hour(0);
    set_minute(0);
    set_second(0);
}

void Clock::sec_to_hms(int n)
{
    int h = (n / 3600);
    int m = (n / 60) - (h * 60);
    int s = n - (m * 60) - (h * 3600);

    h = h % 24;

    set_hour(h);
    set_minute(m);
    set_second(s);
}

int Clock::hms_to_sec()
{
    return (get_hour() * 3600) + (get_minute() * 60) + get_second();
}

int Clock::get_hour() const
{
    return hour;
}

int Clock::get_minute() const
{
    return minute;
}

int Clock::get_second() const
{
    return second;
}
```



```
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;
    }
}

void Clock::increment()
{
    if (second < 59)
    {
        second++;
    }

    else
    {
        second = 0;

        if (minute < 59)
        {
            minute++;
        }

        else
        {
            minute = 0;

            if (hour < 23)
            {
```

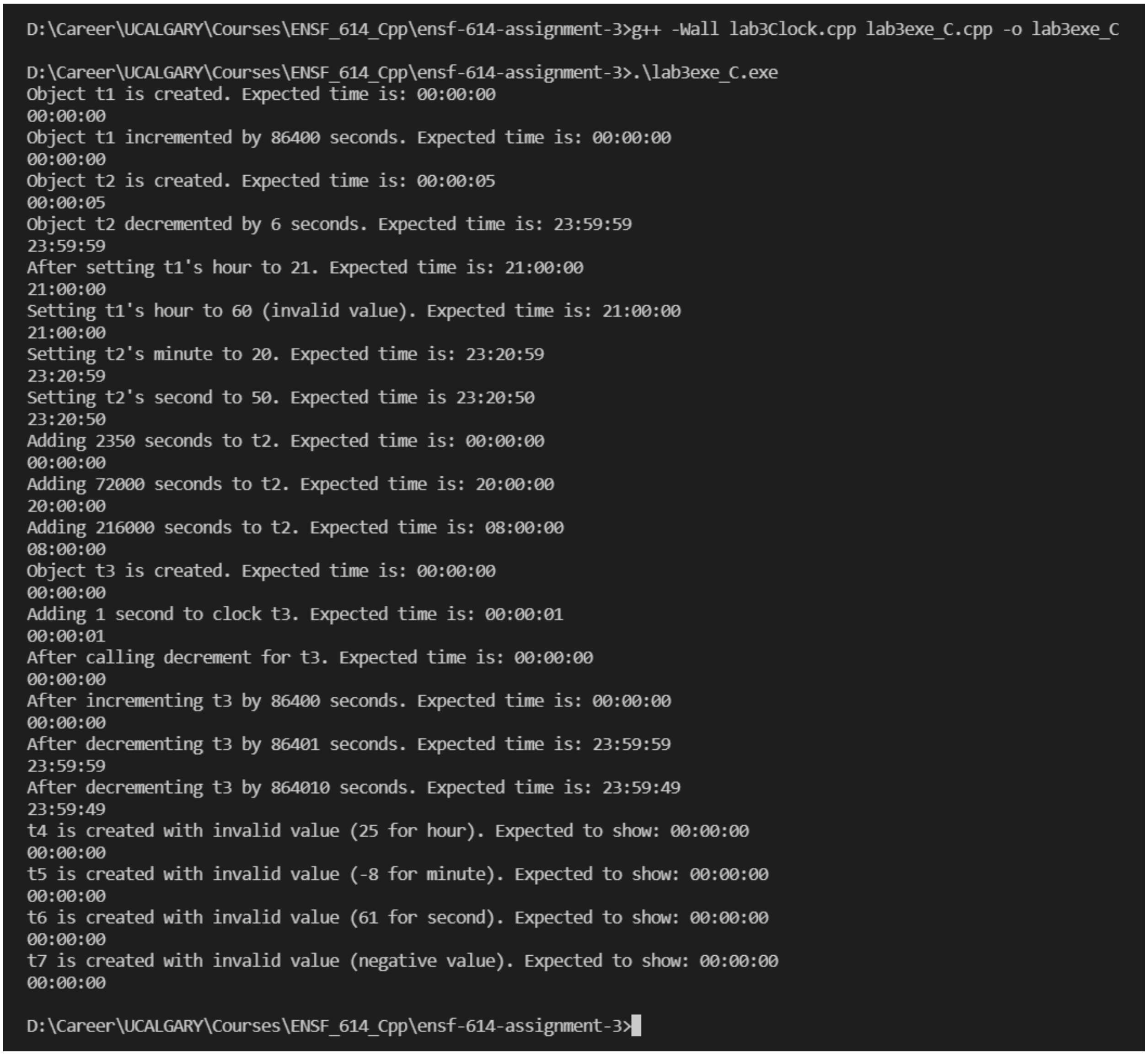


```
                hour++;
            }

            else
            {
                hour = 0;
            }
        }
    }
}

void Clock::decrement()
{
    if (second > 0)
    {
        second--;
    }

    else
    {
        second = 59;

        if (minute > 0)
        {
            minute--;
        }

        else
        {
            minute = 59;

            if (hour > 0)
            {
                hour--;
            }

            else
            {
                hour = 23;
            }
        }
    }
}
```



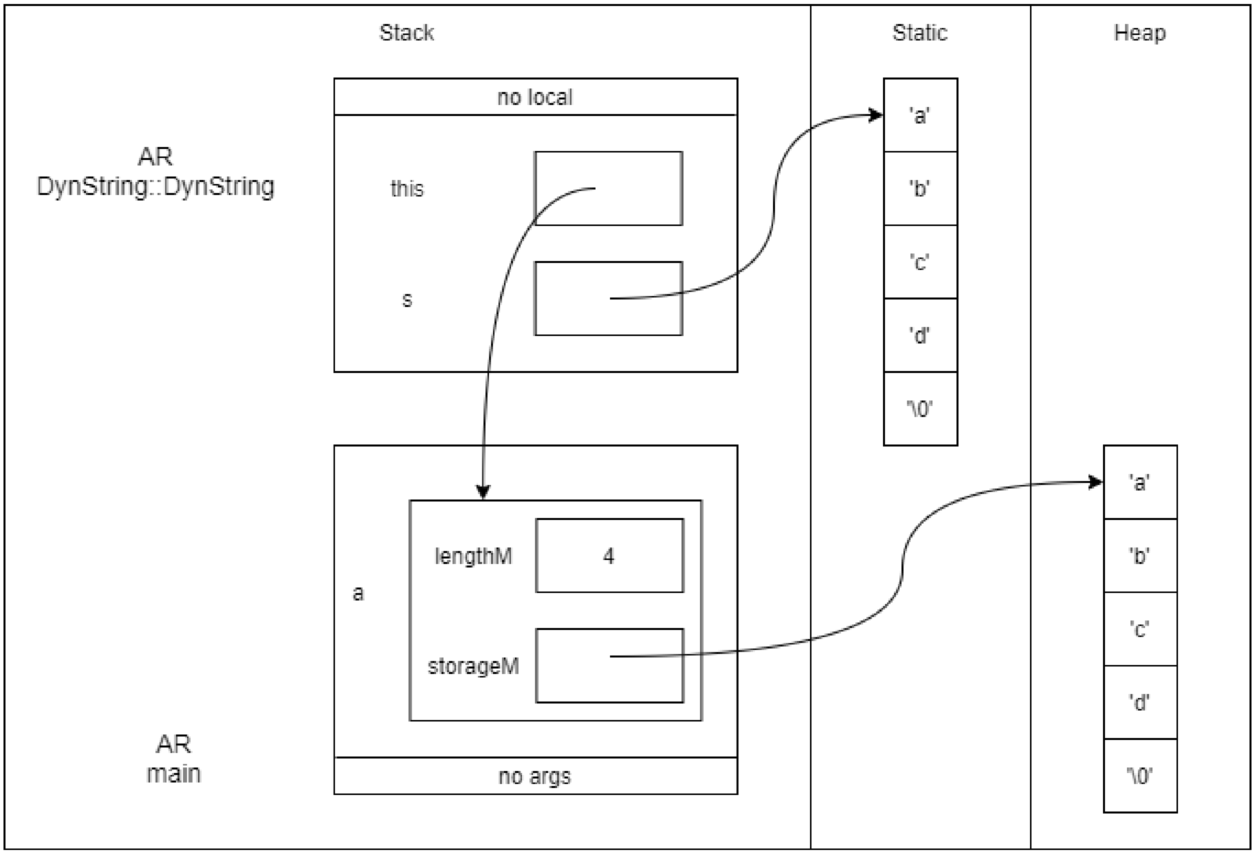
```cpp
void Clock::add_seconds(int s)
{
    for (int i = 0; i < s; i++)
    {
        increment();
    }
}
```

Exercise C – Program Output

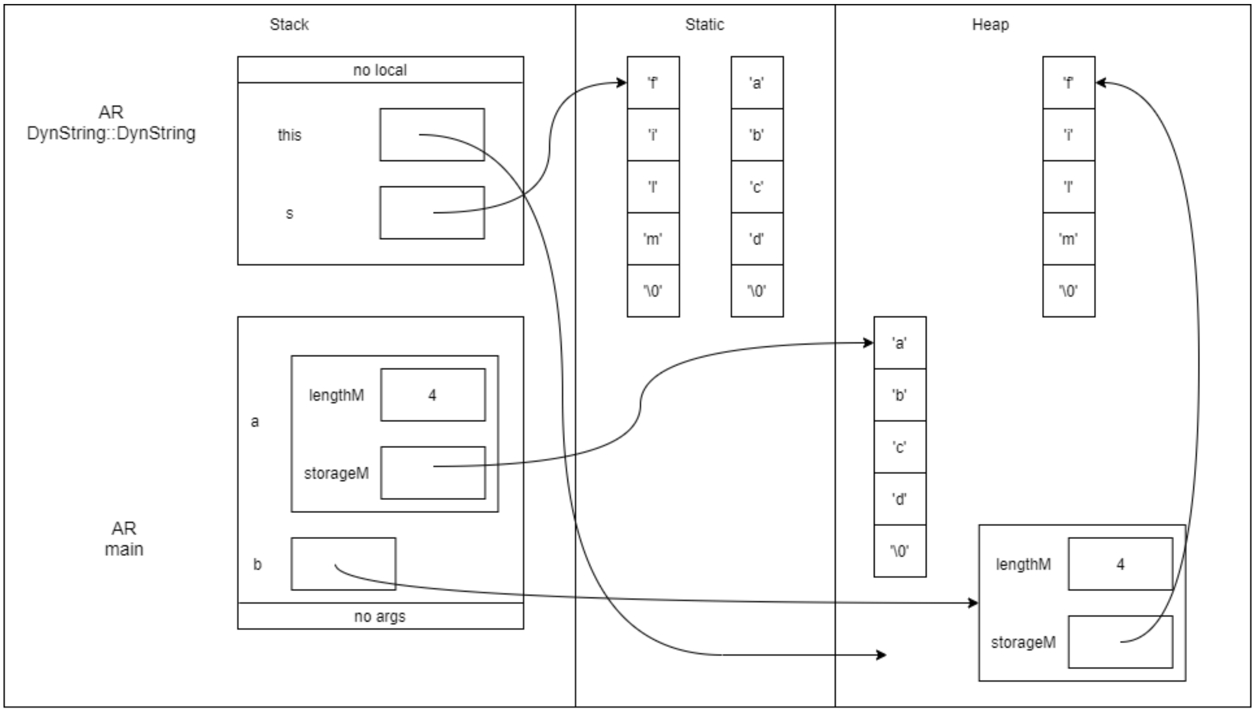


```
D:\Career\UCALGARY\Courses\ENSF_614_Cpp\ensf-614-assignment-3>g++ -Wall lab3Clock.cpp lab3exe_C.cpp -o lab3exe_C

D:\Career\UCALGARY\Courses\ENSF_614_Cpp\ensf-614-assignment-3>.\lab3exe_C.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

D:\Career\UCALGARY\Courses\ENSF_614_Cpp\ensf-614-assignment-3>
```

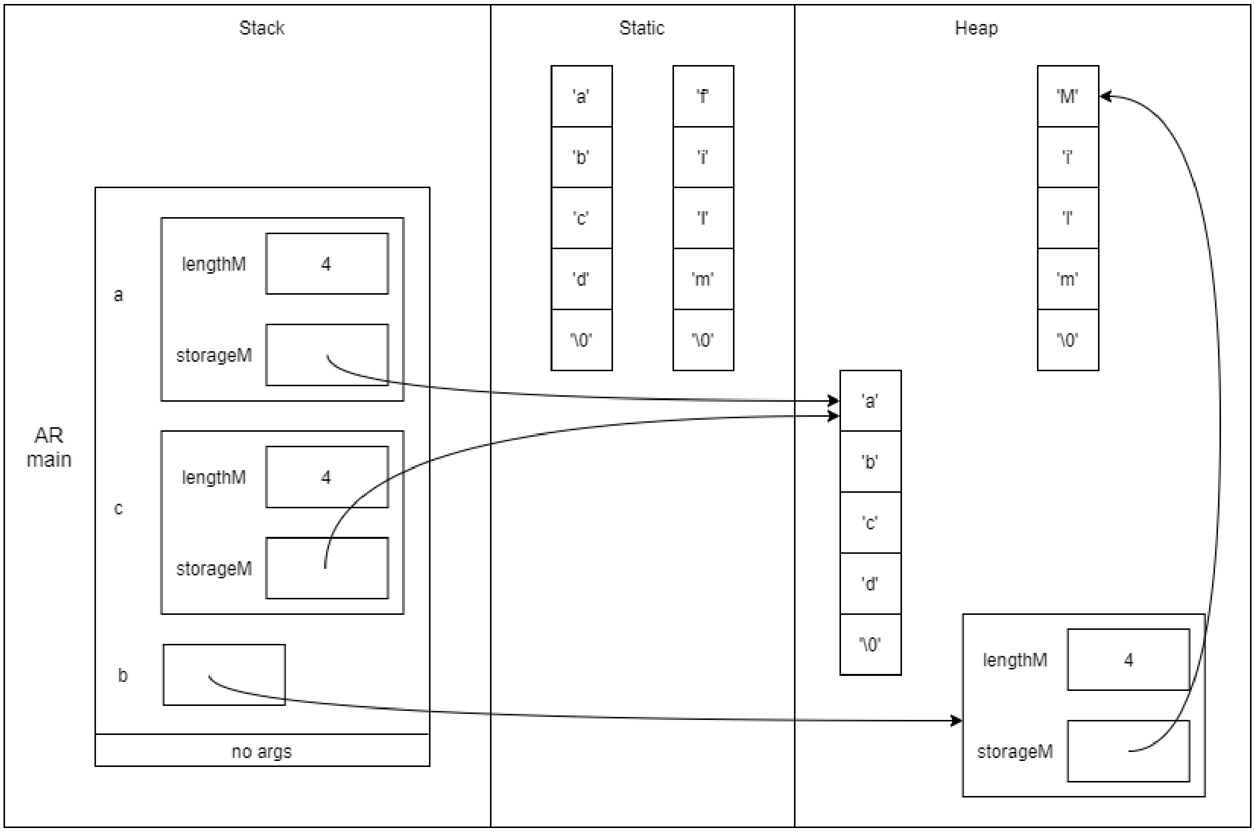
Exercise D – AR Diagram for Point One – First Time



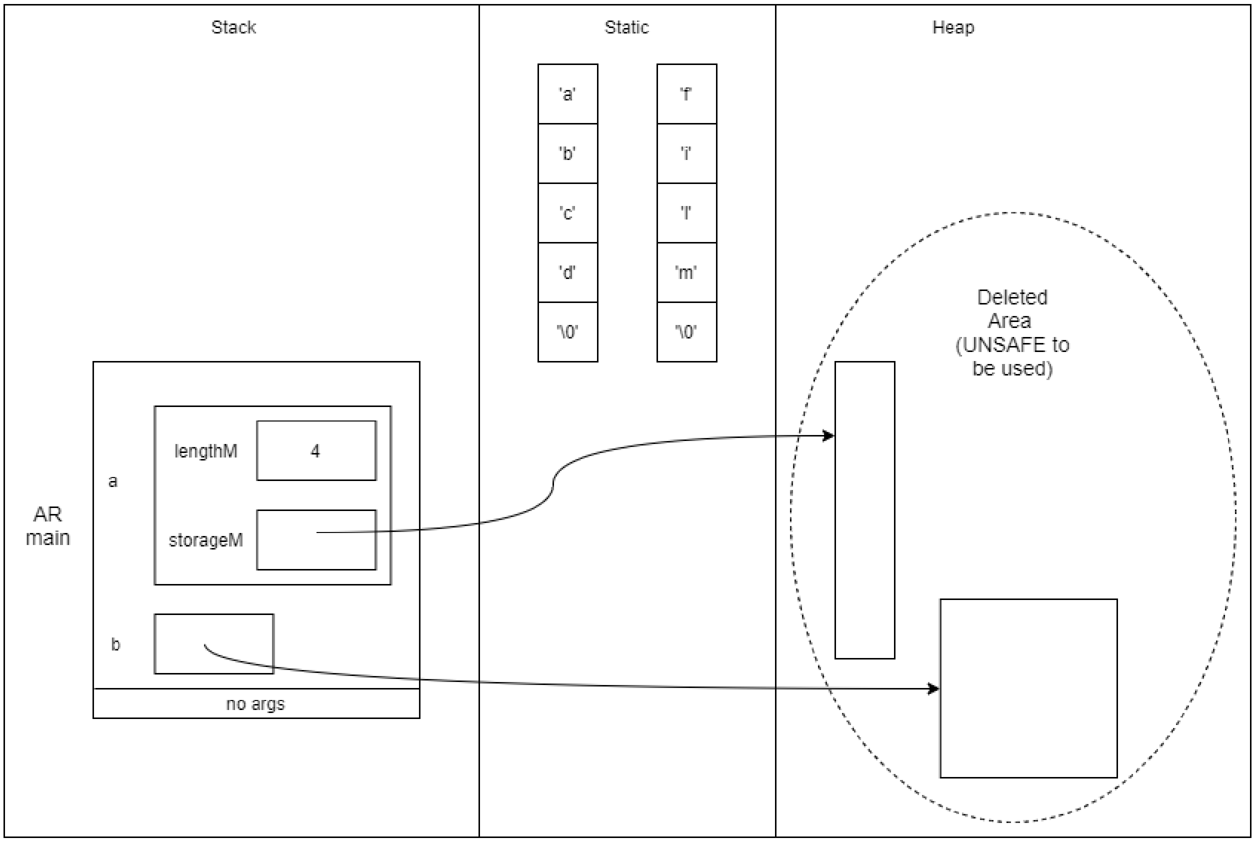
Exercise D – AR Diagram for Point One – Second Time



Exercise D – AR Diagram for Point Three



Exercise D – AR Diagram for Point Four



Exercise D – Answers to questions

1. Till point 4, the constructor of DynString has been called twice.

2. Till point 4, the destructor of DynString has been called twice.

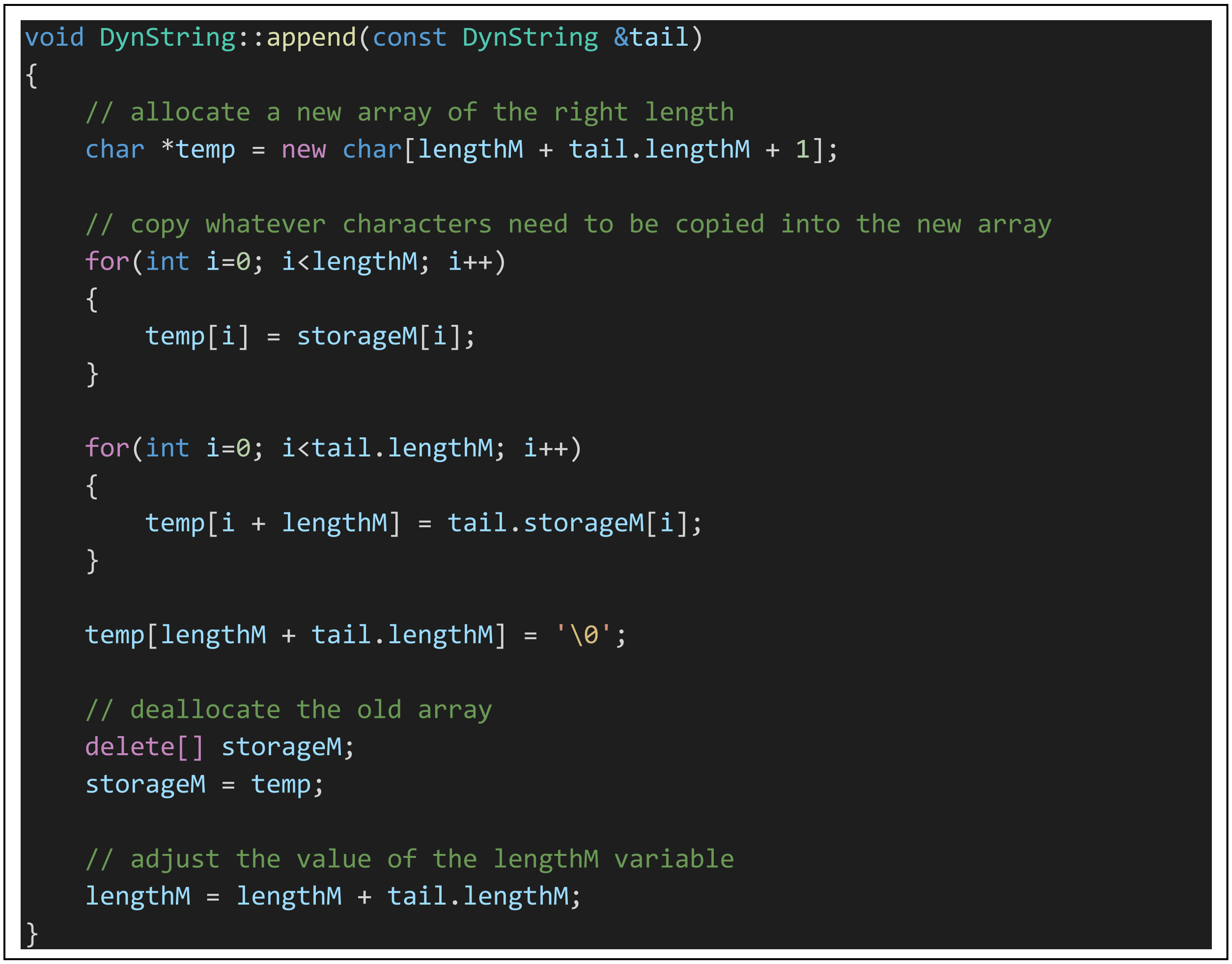
3. Destructor of DynString would be called thrice during the program execution.

4. Object “c” was a shallow copy of object “a”. Hence, the storageM pointer in object “c” pointed to the same location as that of storageM pointer in object “a”.

   When the object “c” goes out of scope during the program, its destructor was called which also deallocates the memory pointed by storageM pointer. This means, the storageM pointer of object “a” is now pointing to de-allocated area on the heap.

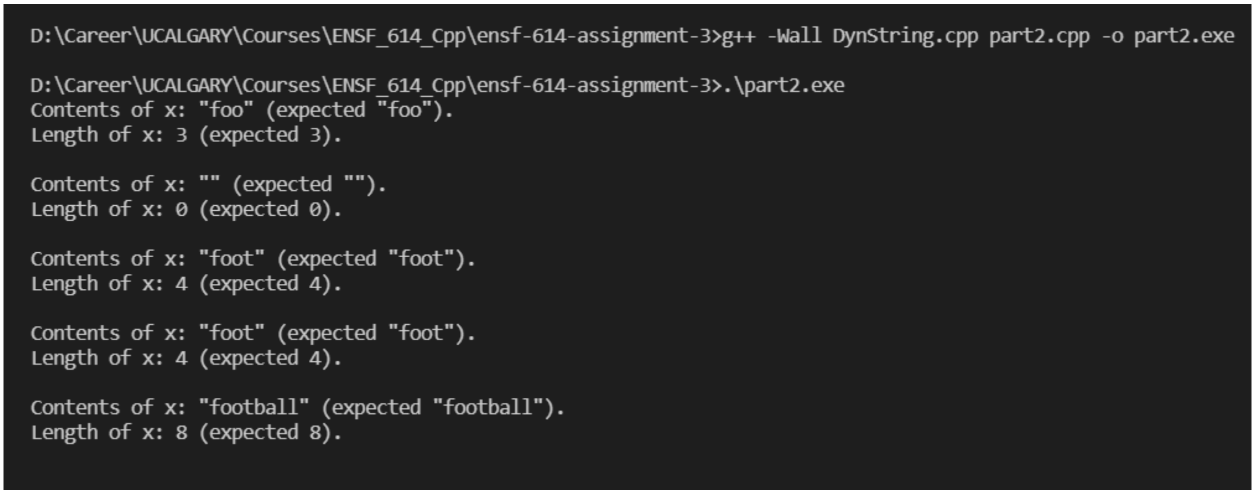
   Just before the program ends, object “a” goes out of scope and its destructor is called. Since memory pointed by storageM is already deleted, when its tried to be deleted again in the destructor, we run into error and the program exits with non-zero return code.

Exercise D – Source Code of append



```
void DynString::append(const DynString &tail)
{
    // allocate a new array of the right length
    char *temp = new char[lengthM + tail.lengthM + 1];

    // copy whatever characters need to be copied into the new array
    for(int i=0; i<lengthM; i++)
    {
        temp[i] = storageM[i];
    }

    for(int i=0; i<tail.lengthM; i++)
    {
        temp[i + lengthM] = tail.storageM[i];
    }

    temp[lengthM + tail.lengthM] = '\0';

    // deallocate the old array
    delete[] storageM;
    storageM = temp;

    // adjust the value of the lengthM variable
    lengthM = lengthM + tail.lengthM;
}
```

Exercise D – Program Output



```
D:\Career\UCALGARY\Courses\ENSF_614_Cpp\ensf-614-assignment-3>g++ -Wall DynString.cpp part2.cpp -o part2.exe

D:\Career\UCALGARY\Courses\ENSF_614_Cpp\ensf-614-assignment-3>.\part2.exe
Contents of x: "foo" (expected "foo").
Length of x: 3 (expected 3).

Contents of x: "" (expected "").
Length of x: 0 (expected 0).

Contents of x: "foot" (expected "foot").
Length of x: 4 (expected 4).

Contents of x: "foot" (expected "foot").
Length of x: 4 (expected 4).

Contents of x: "football" (expected "football").
Length of x: 8 (expected 8).
```