Course : Programming Fundamental – ENSF 337

Lab # : Lab 7

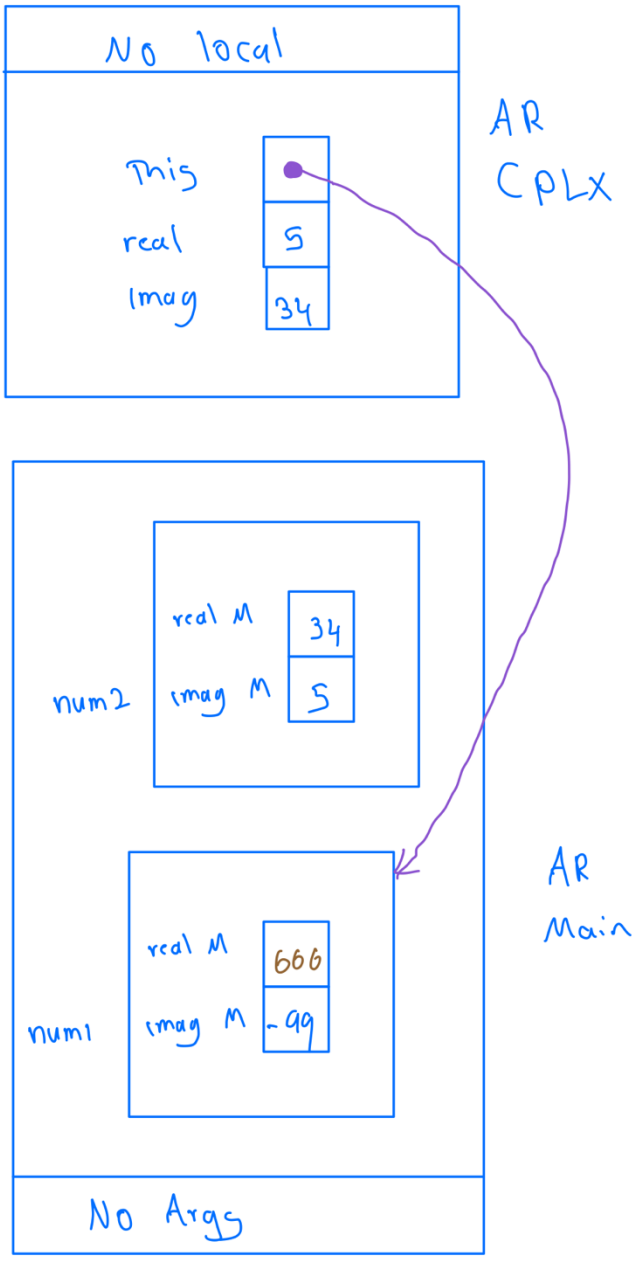
Instructor : M. Moussavi

Student Name : Nimna Wijedasa

Lab Section : B02

Date submitted : Nov 18, 2022

Exercise A


No local
AR
CPLX
This
real
imag
5
34
real M
imag M
num2
34
5
real M
imag M
num1
666
-99
AR
Main
No Args

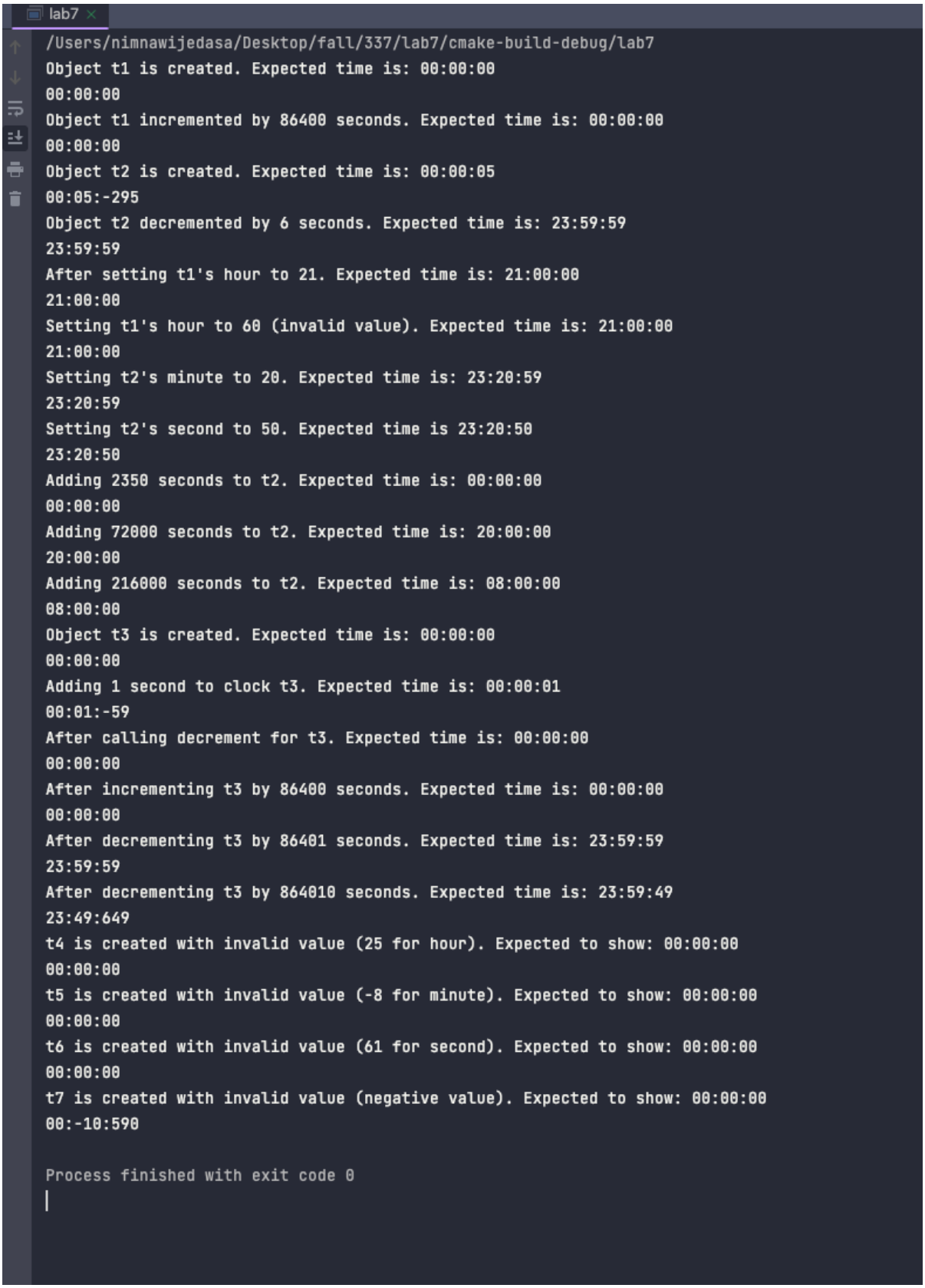
Exercise C

```
#ifndef LAB7_LAB7CLOCK_H
#define LAB7_LAB7CLOCK_H

class Clock {
public:

    Clock();

    explicit Clock (int total_sec);

    Clock (int h, int m, int s);

     int get_hour () const ;
     int get_minute () const ;
     int get_second () const ;


    void set_hour(int arg);
    void set_minute(int arg);
    void set_second(int arg);

    void increment();
    void decrement();
    void add_seconds(int sec);

private:
    int hour{};
    int minute{};
    int second{};
    int hms_to_sec() const;
    void sec_to_hms(int s);

};

#endif //LAB7_LAB7CLOCK_H
```

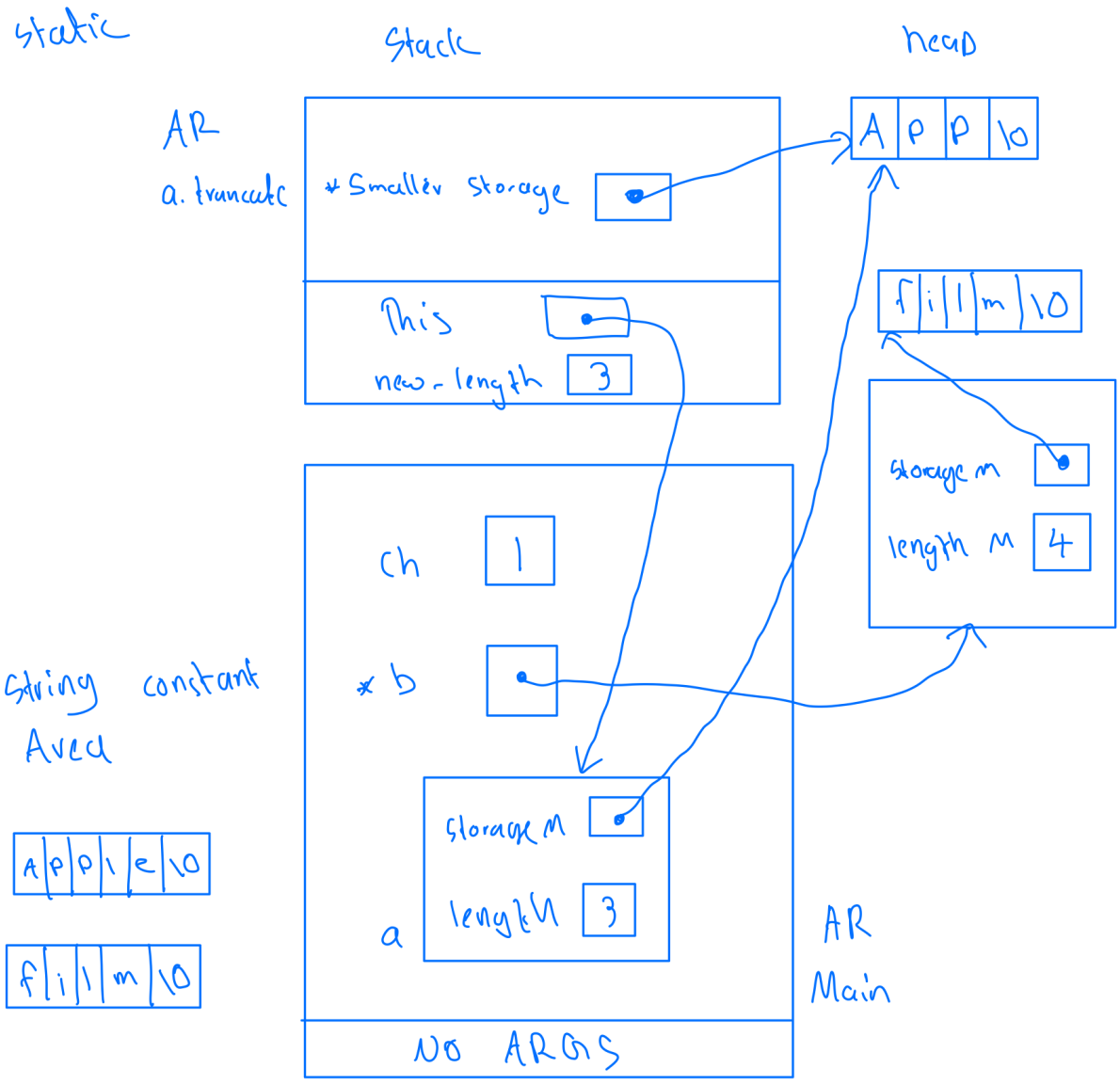
```cpp

#include "lab7Clock.h"

Clock::Clock(): hour(0), minute(0), second(0) {

}

Clock::Clock(int h,int m,int s) {
    if(hour<0 || hour>23 || minute<0 || minute>59 || second<0 || second>59){
        hour = 0;
        minute = 0;
        second = 0;
    }
}

Clock::Clock(int total_sec) {
    if (total_sec < 0){
        hour = 0;
        minute = 0;
        second = 0;
    }
    else{
    }
    sec_to_hms( s: total_sec);
}

int Clock::get_hour() const {
    return hour;
}
int Clock::get_minute() const {
    return minute;
}
int Clock::get_second() const {
    return second;
}
```

```
40
41  void Clock::set_hour(int arg) {
42      if (arg < 23 && arg > 0){
43          hour = arg;
44      }
45  }
46  void Clock::set_minute(int arg) {
47      if (arg < 59 && arg > 0) {
48          minute = arg;
49      }
50  }
51  void Clock::set_second(int arg) {
52      if (arg < 59 && arg > 0) {
53          second = arg;
54      }
55  }
56
57  void Clock::increment() {
58      int sec_total = this->hms_to_sec() + 1;
59      sec_to_hms( s: sec_total);
60  }
61
62  void Clock::decrement() {
63      int sec_total = this->hms_to_sec() - 1;
64      sec_to_hms( s: sec_total);
65
66  }
67
68  void Clock::add_seconds(int sec) {
69      int sec_total = this->hms_to_sec() + sec;
70      sec_to_hms( s: sec_total);
71
72  }
73
```

```cpp
int Clock::hms_to_sec()const {
    int sec_total = hour*3600 + minute*60 + second;
    return sec_total;
}

void Clock::sec_to_hms(int s) {
    s = s % (24*60*60);
    hour = s / 3600;
    minute = s % 60;
    second = s - (hour*3600 + minute*60 );

    if (second == -1)
    {
        second = 59;
        minute--;
    }
    if (minute == -1)
    {
        minute = 59;
        hour--;
    }
    if (hour == -1)
    {
        hour = 23;
    }

}

```


lab7
/Users/nimnawijedasa/Desktop/fall/337/lab7/cmake-build-debug/lab7
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
00:05:-295
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
00:01:-59
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
23:49:649
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:-10:590
Process finished with exit code 0

Exercise D



1) Constructor called 3 times
destructor called 1 time

2) Constructor called 4 times
destructor called 3 time

```
void DynString::append(const DynString& tail)
{
    // Students will complete the definition of this function.
    int tot_len = lengthM + tail.length() +1;

    char* new_arr = new char[tot_len];
    for (int i = 0; i < lengthM ; i++){
        new_arr[i] = storageM[i];
    }
    for (int i = lengthM, j = 0; i < tot_len ; i++, j++){
        new_arr[i] = tail.storageM[j];
    }
    new_arr[tot_len] = '\0';
    delete [] storageM;
    storageM = new_arr;
    lengthM = tot_len - 1;

}
```

```
/Users/nimnawijedasa/Desktop/fall/337/lab7/cmake-build-debug/lab7
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).


Process finished with exit code 0
```

Exercise E

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

//  THIS FUNCTION MUST BE COMPLETED BY THE STUDENTS
if (sizeM == capacityM){
    int updated_cap = (capacityM ==0 ) ? 2 : 2 *capacityM;
    TYPE *new_location = new TYPE [updated_cap];
    for (int i = 0; i < size(); i++)
    {
        new_location[i] = storageM[i];
    }
    delete[ ] storageM;
    storageM = new_location;
}

storageM[sizeM] = val;
sizeM++;
}
```

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


}

SimpleVector& SimpleVector::operator= (const SimpleVector& rhs ){
//  THIS FUNCTION MUST BE COMPLETED BY THE STUDENTS
    int size2 = rhs.size();
    if( this != &rhs){
        TYPE * storage2 = new TYPE [size2];
        for(int i = 0 ; i < size2; i ++){
            storage2[i] = rhs.storageM[i];
        }
        delete [] storageM;
        storageM = storage2;
        sizeM = rhs.size();
        capacityM = rhs.size();
    }
    return *this;
}
```

```
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
-1000 2000 8000

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

Process finished with exit code 0
```