
# CS-110: Lab 09

## One-Dimensional Arrays, Searching and Sorting

https://github.com/mmujtaba25/CS-110

**Muhammad Mujtaba**

**CMD ID: 540040**

mmujtaba.bese25seecs@seecs.edu.pk

**Class:** BESE 16B

**Batch:** 2k25

# Task 1 [CLO 1]:

## CODE:

```cpp
#include <iostream>
#include <random>
#include <iomanip>
#include <sstream>
#include <string>

int getRandomPrintableChar(int minInclusive, int maxInclusive);
std::string getStringOfCharNTimes(char character, size_t times);
void printRow(const std::string &first, const std::string &second, const std::string &third);

int main()
{
    constexpr size_t ARRAY_SIZE = 10;
    int array[ARRAY_SIZE];

    for (size_t i = 0; i < ARRAY_SIZE; i++)
    {
        array[i] = getRandomPrintableChar(1, 100);
    }

    printRow("Element", "Value", "Histogram");
    for (size_t i = 0; i < ARRAY_SIZE; i++)
    {
        printRow(std::to_string(i), std::to_string(array[i]), getStringOfCharNTimes('*',
array[i]));
    }

    std::cout << "\n";
    return 0;
}

int getRandomPrintableChar(int minInclusive, int maxInclusive)
{
    std::random_device rd;
    std::mt19937 gen(rd());
    std::uniform_int_distribution<int> distrib(minInclusive, maxInclusive);
    return distrib(gen);
}

std::string getStringOfCharNTimes(char character, size_t times)
{
    std::stringstream string_;
    for (size_t i = 0; i < times; i++)
    {
        string_ << character;
    }
    return string_.str();
}

void printRow(const std::string &first, const std::string &second, const std::string &third)
{
    constexpr size_t SPACING = 12;
    std::cout << std::left << std::setw(SPACING) << first
              << std::left << std::setw(SPACING) << second
              << std::left << std::setw(SPACING) << third
              << "\n";
}
```

## OUTPUT:

```
obscure@Obscures-MacBook-Air output % ./"task1"
Element     Value       Histogram
0           6           ******
1           64          ****************************************************************
2           69          *********************************************************************
3           61          *************************************************************
4           42          ******************************************
5           88          ****************************************************************************************
6           77          ***************************************************************************
7           63          *****************************************************************
8           42          ******************************************
9           99          *******************************************************************************************************

obscure@Obscures-MacBook-Air output %
```

```
obscure@Obscures-MacBook-Air output % ./"task1"
Element     Value       Histogram
0           70          **********************************************************************
1           72          ************************************************************************
2           43          *******************************************
3           89          ********************************************************************************************
4           6           ******
5           33          *********************************
6           82          *********************************************************************************
7           86          ***********************************************************************************************
8           23          ***********************
9           45          *********************************************

obscure@Obscures-MacBook-Air output %
```

# Task 2 [CLO 2]:

## CODE:

```cpp
#include <iostream>
#include <random>
#include <iomanip>
#include <sstream>
#include <string>
#include <cctype>

#define PRINT_LINE(message) std::cout << message << "\n"
unsigned char getRandomPrintableChar();

int main()
{
    constexpr size_t ARRAY_SIZE = 10;
    unsigned char array[ARRAY_SIZE];

    for (size_t i = 0; i < ARRAY_SIZE; i++)
    {
        array[i] = getRandomPrintableChar();
    }

    int lowerCaseCount = 0;
    int upperCaseCount = 0;
    int digitCount = 0;
    int specialCount = 0;

    for (size_t i = 0; i < ARRAY_SIZE; i++)
    {
        std::cout << array[i] << "_";
        if (islower(array[i]))
            lowerCaseCount++;
        else if (isupper(array[i]))
            upperCaseCount++;
        else if (isdigit(array[i]))
            digitCount++;
        else
            specialCount++;
    }
    std::cout << "\n\n";

    PRINT_LINE(upperCaseCount << " Upper Case Characters Count");
    PRINT_LINE(lowerCaseCount << " Lower Case Characters Count");
    PRINT_LINE(digitCount << " Digits Count");
    PRINT_LINE(specialCount << " Special Characters Count");

    return 0;
}
unsigned char getRandomPrintableChar()
{
    auto getRandomChar = []() -> unsigned char
    {
        // static to reduce overhead by calling continously
        static std::random_device rd;
        static std::mt19937 gen(rd());
        std::uniform_int_distribution<int> distrib(0, 255);
        return distrib(gen);
    };

    unsigned char randomChar = getRandomChar();
    // get a random characgter until
    while (!isprint(randomChar))
    {
        randomChar = getRandomChar();
    }
    return randomChar;
}
```

## OUTPUT:

```
obscure@Obscures-MacBook-Air output % ./"task2"
R_h_v_1_h_V_E_%_W_h_

4 Upper Case Characters Count
4 Lower Case Characters Count
1 Digits Count
1 Special Characters Count
obscure@Obscures-MacBook-Air output %
```

```
obscure@Obscures-MacBook-Air output % ./"task2"
=_{_{_._5_+_:_"_r_C_

1 Upper Case Characters Count
1 Lower Case Characters Count
1 Digits Count
7 Special Characters Count
obscure@Obscures-MacBook-Air output %
```

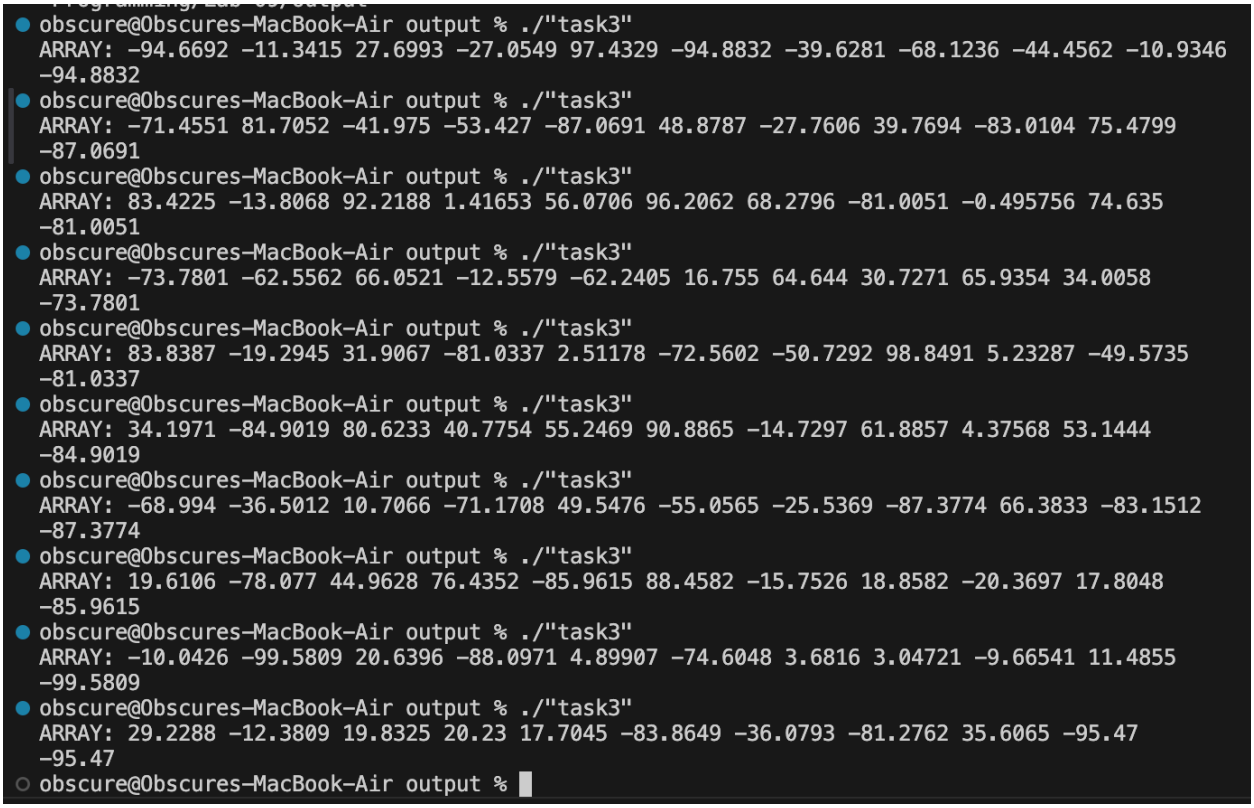
```
obscure@Obscures-MacBook-Air output % ./"task2"
I_;_f_5_L_w_V_I_J_0_

5 Upper Case Characters Count
2 Lower Case Characters Count
2 Digits Count
1 Special Characters Count
obscure@Obscures-MacBook-Air output %
```

# Task 3 [CLO 2]:

## CODE:

```cpp
#include <iostream>
#include <random>

double getRandomNum(double minInclusive, double maxInclusive);
double min(double array[], size_t size);

int main()
{
    constexpr size_t TEST_ARRAY_SIZE = 10;
    double testArray[TEST_ARRAY_SIZE];

    for (size_t i = 0; i < TEST_ARRAY_SIZE; i++)
    {
        testArray[i] = getRandomNum(-100, 100);
    }

    std::cout << "ARRAY: ";
    for (size_t i = 0; i < TEST_ARRAY_SIZE; i++)
    {
        std::cout << testArray[i] << " ";
    }
    std::cout << "\n";

    std::cout << min(testArray, TEST_ARRAY_SIZE) << "\n";

    // std::cin.get();
    return 0;
}

double getRandomNum(double minInclusive, double maxInclusive)
{
    if (minInclusive > maxInclusive)
        std::swap(minInclusive, maxInclusive);

    static std::random_device rd;
    static std::mt19937 gen(rd());
    std::uniform_real_distribution<double> distribution(minInclusive, maxInclusive);
    return distribution(gen);
}

double min(double array[], size_t size)
{
    double min = array[0];
    for (size_t i = 0; i < size; i++)
    {
        if (array[i] < min)
        {
            min = array[i];
        }
    }

    return min;
}
```

## OUTPUT:


obscure@Obscures-MacBook-Air output % ./"task3"
ARRAY: -94.6692 -11.3415 27.6993 -27.0549 97.4329 -94.8832 -39.6281 -68.1236 -44.4562 -10.9346
-94.8832
obscure@Obscures-MacBook-Air output % ./"task3"
ARRAY: -71.4551 81.7052 -41.975 -53.427 -87.0691 48.8787 -27.7606 39.7694 -83.0104 75.4799
-87.0691
obscure@Obscures-MacBook-Air output % ./"task3"
ARRAY: 83.4225 -13.8068 92.2188 1.41653 56.0706 96.2062 68.2796 -81.0051 -0.495756 74.635
-81.0051
obscure@Obscures-MacBook-Air output % ./"task3"
ARRAY: -73.7801 -62.5562 66.0521 -12.5579 -62.2405 16.755 64.644 30.7271 65.9354 34.0058
-73.7801
obscure@Obscures-MacBook-Air output % ./"task3"
ARRAY: 83.8387 -19.2945 31.9067 -81.0337 2.51178 -72.5602 -50.7292 98.8491 5.23287 -49.5735
-81.0337
obscure@Obscures-MacBook-Air output % ./"task3"
ARRAY: 34.1971 -84.9019 80.6233 40.7754 55.2469 90.8865 -14.7297 61.8857 4.37568 53.1444
-84.9019
obscure@Obscures-MacBook-Air output % ./"task3"
ARRAY: -68.994 -36.5012 10.7066 -71.1708 49.5476 -55.0565 -25.5369 -87.3774 66.3833 -83.1512
-87.3774
obscure@Obscures-MacBook-Air output % ./"task3"
ARRAY: 19.6106 -78.077 44.9628 76.4352 -85.9615 88.4582 -15.7526 18.8582 -20.3697 17.8048
-85.9615
obscure@Obscures-MacBook-Air output % ./"task3"
ARRAY: -10.0426 -99.5809 20.6396 -88.0971 4.89907 -74.6048 3.6816 3.04721 -9.66541 11.4855
-99.5809
obscure@Obscures-MacBook-Air output % ./"task3"
ARRAY: 29.2288 -12.3809 19.8325 20.23 17.7045 -83.8649 -36.0793 -81.2762 35.6065 -95.47
-95.47
obscure@Obscures-MacBook-Air output %

# Task 4 [CLO 1]:

## CODE:

```cpp
#include <iostream>
#include <random>
#include <iomanip>

int getRandomNum(int minInclusive, int maxInclusive);
void sortArrayInPlace(int array[], const int size);
void printArray(int array[], const size_t size);

int main()
{
    constexpr size_t TEST_ARRAY_SIZE = 10;
    int testArray[TEST_ARRAY_SIZE];

    for (size_t i = 0; i < TEST_ARRAY_SIZE; i++)
    {
        testArray[i] = getRandomNum(-100, 100);
    }

    std::cout << "BEFORE SORT: ";
    printArray(testArray, TEST_ARRAY_SIZE);

    sortArrayInPlace(testArray, TEST_ARRAY_SIZE);
    std::cout << "AFTER  SORT: ";
    printArray(testArray, TEST_ARRAY_SIZE);

    // std::cin.get();
    return 0;
}

void printArray(int array[], const size_t size)
{
    for (size_t i = 0; i < size; i++)
    {
        std::cout << std::right << std::setw(5) << array[i];
    }
    std::cout << "\n";
}

int getRandomNum(int minInclusive, int maxInclusive)
{
    if (minInclusive > maxInclusive)
        std::swap(minInclusive, maxInclusive);

    static std::random_device rd;
    static std::mt19937 gen(rd());
    std::uniform_int_distribution<int> distribution(minInclusive, maxInclusive);
    return distribution(gen);
}

void sortArrayInPlace(int array[], int size)
{
    // use bubble sort algorithm
    int i, j;
    for (j = 0; j < size; j++)
    {
        for (i = 0; i < (size - 1); i++)
        {
            if (array[i] > array[i + 1])
            {
                std::swap(array[i], array[i + 1]);
            }
        }
    }
}
```

## OUTPUT:

```
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:   -42  -92    1    9   97    9    1  -37 -100  -28
AFTER  SORT:  -100  -92  -42  -37  -28    1    1    9    9   97
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:    96  -26   -3   94   59   89   16  -92   96  -54
AFTER  SORT:   -92  -54  -26   -3   16   59   89   94   96   96
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:   -91   87   94  -16   23   25   81  -45   91   32
AFTER  SORT:   -91  -45  -16   23   25   32   81   87   91   94
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:   -86   20   63   78  -40   90    1  -20  -83   38
AFTER  SORT:   -86  -83  -40  -20    1   20   38   63   78   90
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:    99   70   25   -2  -43  -14  -77  -95   94   89
AFTER  SORT:   -95  -77  -43  -14   -2   25   70   89   94   99
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:    98  -82   -2   73   -5   65  -86   26   44  -84
AFTER  SORT:   -86  -84  -82   -5   -2   26   44   65   73   98
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:   -67  -19  -75   64  -57   53   13   42    3  -68
AFTER  SORT:   -75  -68  -67  -57  -19    3   13   42   53   64
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:   -80   11  -15  -21   77  -50  -44  -98   94   -3
AFTER  SORT:   -98  -80  -50  -44  -21  -15   -3   11   77   94
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:   -74   40   -6   33   -9   78  -17  -68   10   38
AFTER  SORT:   -74  -68  -17   -9   -6   10   33   38   40   78
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:   -31  -37   92  -87   -1  -92   85  -85  -39   47
AFTER  SORT:   -92  -87  -85  -39  -37  -31   -1   47   85   92
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:   -73   91  -28  -26   64   14   97  -90  -73  -82
AFTER  SORT:   -90  -82  -73  -73  -28  -26   14   64   91   97
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:    16  -66  -91  -59   23   70  -74   17  -26   33
AFTER  SORT:   -91  -74  -66  -59  -26   16   17   23   33   70
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:   -56  -30   22   38  -65  -78  -81  -54   16   46
AFTER  SORT:   -81  -78  -65  -56  -54  -30   16   22   38   46
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:   -96   29  -85   52  -25  -39  -41  -74  -33   67
AFTER  SORT:   -96  -85  -74  -41  -39  -33  -25   29   52   67
obscure@Obscures-MacBook-Air output % ./"task4"
BEFORE SORT:   -27   98  -16   56  -74   94  -47   48   67  -42
AFTER  SORT:   -74  -47  -42  -27  -16   48   56   67   94   98
obscure@Obscures-MacBook-Air output %
obscure@Obscures-MacBook-Air output %
```

# Task 5 [CLO 1]:

## CODE:

```
#include <iostream>
#include <random>
#include <iomanip>

int getRandomNum(int minInclusive, int maxInclusive);
void sortArrayInPlace(int array[], const int size);
void printArray(int array[], const size_t size);

int getElementIndex(int array[], const size_t size, int element);

int main()
{
    constexpr size_t TEST_ARRAY_SIZE = 100;
    int testArray[TEST_ARRAY_SIZE];

    for (size_t i = 0; i < TEST_ARRAY_SIZE; i++)
    {
        testArray[i] = getRandomNum(-999, 999);
    }

    sortArrayInPlace(testArray, TEST_ARRAY_SIZE);

    std::cout << "SORTED ARRAY:\n";
    printArray(testArray, TEST_ARRAY_SIZE);
    std::cout << "\n";

    int toFound;
    std::cout << "Enter element to find from array: ";
    std::cin >> toFound;

    std::cout << "[" << toFound << "] at index " << getElementIndex(testArray, TEST_ARRAY_SIZE,
toFound) << "\n";

    // std::cin.get();
    return 0;
}

void printArray(int array[], const size_t size)
{
    constexpr size_t COLUMNS = 5;
    for (size_t i = 0; i < size; i++)
    {
        std::cout << "[" << std::setw(4) << std::right << array[i] << " @ " << std::setw(4) <<
std::left << i << "] ";
        if (i % COLUMNS == COLUMNS - 1)
        {
            std::cout << "\n";
        }
    }
    std::cout << "\n";
}

int getRandomNum(int minInclusive, int maxInclusive)
{
    if (minInclusive > maxInclusive)
        std::swap(minInclusive, maxInclusive);

    static std::random_device rd;
    static std::mt19937 gen(rd());
    std::uniform_int_distribution<int> distribution(minInclusive, maxInclusive);
    return distribution(gen);
}
```

```cpp
void sortArrayInPlace(int array[], int size)
{
    // use bubble sort algorithm
    int i, j;
    for (j = 0; j < size; j++)
    {
        for (i = 0; i < (size - 1); i++)
        {
            if (array[i] > array[i + 1])
            {
                std::swap(array[i], array[i + 1]);
            }
        }
    }
}

int getElementIndex(int array[], const size_t size, int element)
{
    int min = 0;
    int max = static_cast<int>(size) - 1;
    int firstIndex = -1;

    while (min <= max)
    {
        int mid = (min + max) / 2;
        if (array[mid] > element)
            max = mid - 1;
        else if (array[mid] < element)
            min = mid + 1;
        else
        {
            firstIndex = mid;
            max = mid - 1; // force left search
        }
    }

    // if not found return -1
    return firstIndex;
}
```

## OUTPUT:

```
obscure@Obscures-MacBook-Air output % ./"task5"
SORTED ARRAY:
[-967 @ 0   ] [-966 @ 1   ] [-945 @ 2   ] [-945 @ 3   ] [-904 @ 4   ]
[-904 @ 5   ] [-888 @ 6   ] [-859 @ 7   ] [-821 @ 8   ] [-818 @ 9   ]
[-816 @ 10  ] [-769 @ 11  ] [-751 @ 12  ] [-705 @ 13  ] [-655 @ 14  ]
[-636 @ 15  ] [-627 @ 16  ] [-614 @ 17  ] [-607 @ 18  ] [-579 @ 19  ]
[-561 @ 20  ] [-552 @ 21  ] [-549 @ 22  ] [-546 @ 23  ] [-523 @ 24  ]
[-508 @ 25  ] [-462 @ 26  ] [-459 @ 27  ] [-444 @ 28  ] [-436 @ 29  ]
[-325 @ 30  ] [-284 @ 31  ] [-275 @ 32  ] [-218 @ 33  ] [-165 @ 34  ]
[-163 @ 35  ] [-141 @ 36  ] [-141 @ 37  ] [-122 @ 38  ] [-115 @ 39  ]
[ -95 @ 40  ] [ -82 @ 41  ] [ -67 @ 42  ] [ -58 @ 43  ] [ -55 @ 44  ]
[ -43 @ 45  ] [  13 @ 46  ] [  24 @ 47  ] [  44 @ 48  ] [  50 @ 49  ]
[  77 @ 50  ] [  79 @ 51  ] [  83 @ 52  ] [  95 @ 53  ] [ 187 @ 54  ]
[ 195 @ 55  ] [ 205 @ 56  ] [ 224 @ 57  ] [ 231 @ 58  ] [ 234 @ 59  ]
[ 237 @ 60  ] [ 252 @ 61  ] [ 266 @ 62  ] [ 272 @ 63  ] [ 273 @ 64  ]
[ 284 @ 65  ] [ 352 @ 66  ] [ 354 @ 67  ] [ 363 @ 68  ] [ 367 @ 69  ]
[ 383 @ 70  ] [ 445 @ 71  ] [ 466 @ 72  ] [ 498 @ 73  ] [ 517 @ 74  ]
[ 540 @ 75  ] [ 630 @ 76  ] [ 633 @ 77  ] [ 665 @ 78  ] [ 669 @ 79  ]
[ 689 @ 80  ] [ 748 @ 81  ] [ 755 @ 82  ] [ 766 @ 83  ] [ 778 @ 84  ]
[ 787 @ 85  ] [ 801 @ 86  ] [ 826 @ 87  ] [ 828 @ 88  ] [ 843 @ 89  ]
[ 852 @ 90  ] [ 865 @ 91  ] [ 866 @ 92  ] [ 869 @ 93  ] [ 940 @ 94  ]
[ 944 @ 95  ] [ 952 @ 96  ] [ 967 @ 97  ] [ 981 @ 98  ] [ 993 @ 99  ]


Enter element to find from array: -945
[-945] at index 2
obscure@Obscures-MacBook-Air output %
```