CS111 Lab 20 – 21 Solutions

Goal: Understand 1D array and get familiar with different types of questions about array in mid-term.

1) Provide the output of the given line in the blanks below.

```cpp
#include <iostream>
using namespace std;

int main() {
    int a[10] = {1, 6, 4, 2, 8, 4, 1, 3, 2, -2};
    cout << a[0] << endl; // (1) 1
    cout << a[3] << endl; // (2) 2
    cout << a[5] << endl; // (3) 4
    cout << a[9] << endl; // (4) -2

    for (int i = 0; i < 10; ++i)
        a[i] += i;

    cout << a[1] << endl; // (5) 7
    cout << a[2] << endl; // (6) 6
    cout << a[6] << endl; // (7) 7
    cout << a[8] << endl; // (8) 10
    return 0;
}
```
Writing a program:
2) Write a complete C++ program which carries out the following tasks:
   - Declares an integer array of size 10.
   - Accepts 10 inputs from a user (to be stored in the array).
   - Squares each element of the array.
   - Prints the array elements.

#include <iostream>
using namespace std;

int main() {
    int inputs[10];

    //input
    for (int i = 0; i < 10; ++i) {
        cout << "Enter input #" << i+1 << ": ";
        cin >> inputs[i];
    }

    //squaring
    for (int i = 0; i < 10; ++i) {
        inputs[i] = inputs[i] * inputs[i];
    }

    //printing
    cout << "Your array now: ";
    for (int i = 0; i < 10; ++i) {
        cout << inputs[i] << " ";
    }
    cout << endl;

    return 0;
}
3) Write a complete C++ program which carries out the following tasks:
   a. Declares an integer array of size 10.
   b. Fills the array with random numbers between 1 and 100.
   c. Prints the array elements.
   d. Gets and prints the maximum number in the array.

```cpp
#include <iostream>
#include <cstdlib>
#include <ctime>

using namespace std;

int main() {
    srand(time(0));
    int a[10];

    for (int i = 0; i < 10; ++i) {
        a[i] = rand() % 100 + 1;
    }

    for (int i = 0; i < 10; ++i) {
        cout << a[i] << " ";
    }
    cout << endl;

    int max = a[0]; // assume first # to be max
    for (int i = 1; i < 10; ++i) {
        if (a[i] > max) max = a[i];
    }

    cout << "Max number: " << max << endl;
    return 0;
}
```
Title Lines:

4) (prac2.pdf) Write **title lines** for the functions that are called by the following main program. **Do not supply the blocks for the functions.**

```c
int main() {
    int a[5] = {3,1,4,1,5};
    printAverage(a, 9); // prints average
    swap(a, 3, 2); // swap elements 3 and 2
    reverse(a[1]); // reverse the digits in a[1]
    if (isPositive(a[0])) cout << "Positive" << endl;
    cout << midEntry(a, 5) << endl; // prints: Positive
    return 0;
}
```

a) Title line for **printAverage**
   
   ```c
   void printAverage (int a [], int cap)
   ```

b) Title line for **swap**
   
   ```c
   void swap(int a[], int i, int j )
   ```

c) Title line for **reverse**
   
   ```c
   void reverse(int &a)
   ```

d) Title line for **isPositive**
   
   ```c
   bool isPositive(int x)
   ```

e) Title line for **midEntry**
   
   ```c
   int midEntry(int a[], int cap)
   ```

5) (prac2.pdf) Write **title lines** (header lines or prototypes) for the following functions. Do not supply the blocks for the functions.

(a) A function called **add3** which returns the sum of three double parameters.

   ```c
   double add3 (double a, double b, double c)
   ```

(b) A function called **reverseIt** that returns the number obtained by reversing the digits in an integer parameter.

   ```c
   int reverseIt (int x)
   ```

c) A function called **randomArray** that sets the values in an array of doubles to have random values.

   ```c
   void randomArray (double arr [], int capacity)
   ```

d) A function called **addTwo** that adds 2 to every entry in an array of integers.

   ```c
   void addTwo(int array[], int cap)
   ```

e) A function called **biggerAverage** which determines which of two arrays of integers has the bigger average. It should return the value of this bigger average.

   ```c
   double biggerAverage(int array1[], int cap1, int array2[], int cap2)
   ```
Tracing through a program for output:
6) (prac2.pdf) Consider the following C++ program.

```cpp
#include <iostream>
using namespace std;

void mystery(int data[], int p, int q) {
    data[p] = data[q];
    data[q] = data[p];
}

void m2(int &p, int q) {
    int temp = p;
    p = q;
    q = temp;
}

void print(int data[], int p) {
    for (int i = 0; i < p; i++)
        cout << data[i] << " ";
    cout << endl;
}

int main() {
    int x[8] = {0, 1, 2, 3, 4, 5, 6, 7};
    int y[7] = {0, 1, 2, 3, 4, 5, 6};
    int a = 3, b = 4;

    print(x, 3);       // line (a)
    mystery(x, 1, 2); print(x, 5);   // line (b)
    for (int i = 1; i <= 7; i++) mystery(x, 0 ,i);
    print(x, 8);       // line (c)
    m2(a, b); cout << a << b << endl;   // line (d)
    m2(y[3], 7); print(y, 6);    // line (e)
    return 0;
}
```

(a) What is the output at line (a)? 0 1 2
(b) What is the output at line (b)? 0 2 2 3 4
(c) What is the output at line (c)? 7 2 2 3 4 5 6 7
(d) What is the output at line (d)? 44
(e) What is the output at line (e)? 0 1 2 7 4 5
7) (prac3.pdf) Consider the following C++ program.

```cpp
#include <iostream>
using namespace std;

int recursive (int x) {
    if (x < 5) return 3;
    return recursive (x / 3) + x % 6;
}
char swap (int x, int y) {
    x = y;
    y = x;
    cout << x << y;
    return 's';
}
void set (int arr []) {
}

int main() {
    int x[5];
    set(x);
    swap(1, 2); cout << endl;       //line (a)
    set(x);
    cout << x[0 + 2] << x[0] + 2 << endl;    //line (b)
    cout << swap(1, 2) << endl;       //line (c)
    for (int i = 1; i < 4; i++) cout << x[i]; cout << endl;  //line (d)
    int e = 21;
    cout << recursive(e) << endl;      //line (e)
    return 0;
}
```

(a) What is the output at line (a)? 22
(b) What is the output at line (b)? 63
(c) What is the output at line (c)? 22s
(d) What is the output at line (d)? 968
(e) What is the output at line (e)? 7
Short Blocks of code:
8) (prac3.pdf) Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```cpp
int main() {
    int i = 2;
    int x[5] = {3, 1, 4, 1, 5};
    // (a) Return the sum. Here 4 is printed.
    cout << add(i, 2) << endl;
    // (b) Return number of odd entries. Here 4 is printed.
    cout << numOdd(x, 5) << endl;
    // (c) Multiply i by 2. Here 4 is printed.
    doubleIt(i); cout << i << endl;
    // (d) Find the index of the largest entry. Here 4 is printed.
    cout << findIndexMax(x, 5) << endl;
    // (e) Return the absolute value. Here 4 is printed.
    cout << absoluteValue(i) << endl;
    return 0;
}

(a)
int add(int x, int y) {
    return x + y;
}

(b)
int numOdd(int array[], int cap) {
    int ans = 0;
    for (int i = 0; i < cap; i++)
        if (array[i] % 2 != 0) ans++;
    return ans;
}

(c)
void doubleIt(int &x) {
    x *= 2;
}

(d)
int findIndexMax(int array[], int cap) {
    int ans = 0;
    for (int i = 0; i < cap; i++)
        if (array[i] > array[ans]) ans = i;
    return ans;
}

(e)
int absoluteValue(int x) {
    if (x < 0) return -x;
    return x;
}
```
9) (prac3.pdf) Write blocks of code to perform the functions used in the following main program. Your blocks must match the given title lines. Each block should be a short function of only a few lines.

```cpp
int main() {
    int x = 5;
    double e = 2.718;
    double a[4] = {1.0, 2.0, -3.0, -4.0};
    double b[2] = {5.5, 4.5};
    // (a) Changes the sign. Here to -5
    changeSign(x);
    // (b) Return first digit after decimal point. Here 7 is printed.
    cout << firstDecimal(e) << endl;
    // (c) Return the number of negative entries. Here 2 is printed.
    cout << numberNeg(a, 4) << endl;
    // (d) Test whether the first argument is a factor of the second. Here: Yes
    if (isFactor(7, 14)) cout << "Yes\n";
    // (e) print average of all entries both arrays: Here 1.0 is printed.
    averageArrays(a, 4, b, 2);
    return 0;
}

(a)
void changeSign(int &x) {
    x = -x;
}

(b)
int firstDecimal(double x) {
    int tenX = (int) (x * 10);
    return tenX % 10;
}

(c)
int numberNeg(double x[], int capacity) {
    int ans = 0;
    for (int i = 0; i < capacity; i++)
        if (x[i] < 0) ans++;
    return ans;
}

(d)
bool isFactor(int x, int y) {
    return y % x == 0;
}

(e)
void averageArrays(double x[], int capacityX, double y[], int capacityY) {
    double sum = 0.0;
    for (int i = 0; i < capacityX; i++) sum += x[i];
    for (int i = 0; i < capacityY; i++) sum += y[i];
    cout << sum / (capacityX + capacityY) << endl;
}
```