**LAB 2 Data structures and Algorithms:**

**Muhammad Ahmed Fraz**

**332779**

**BSCS-10C**

**Task1:**

#include<iostream>

using namespace std;

int main() {

int\* salary = new int[20];

//Initializing dynamic array

int i;

for (i = 0; i < 20; ++i)

{

cout << "Enter Salary: ";

cin >> \*(salary+i);

//Taking the input

}

for (i = 0; i < 20; ++i)

\*(salary+i) = \*(salary+i) + \*(salary+i) / (i + 1);

delete[] salary;

//Deleting the dynamic array from the heap

return 0;

}

**Task 2**

Complete the two parts for analyze pointer problem in the Lab\_2\_-\_Problem\_2.cpp file.

Part 1: Write a function void analyze\_pointer(int \*ptr) that does two things:

* Write the memory location pointed by the pointer to the console.
* Write the value of the integer (which the pointer points to) to the console.

void analyze\_pointer(int\* ptr)

{

// Add your code here!

cout << "The memory location pointed by the pointer to the console is: " << ptr << endl;

cout << "The value of the integer which the pointer points to the console is: " << \*(ptr);

}

Part 2: Use the function to complete two tasks:

* Allocate an int on the stack (e.g., “int iValue;"), assign a value to it, and get its memory location (with the reference operator—&) to pass this value to analyze\_pointer.
* Allocate an int on the heap (with the new operator). Assign a value to it, and pass it to analyze\_pointer.

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

**Task 3**

Define a struct Area that has two private variable members; units of type string and area\_value of type float. Modify the Lab\_2\_-\_Problem\_3.cpp program to create a dynamic variable of type Area.

* Input from the keyboard the area\_value and its units. Compute one-half and one quarter of the area and display the results
* Destroy the dynamic variable at the end

Code:

/\*

\* Kindly fill this information.

\* Name: Muhammad Ahmed Fraz

\* Student #: 332779

\* Date: 12/09/13

\*/

#include <iostream>

#include <string>

using namespace std;

/\*

\* Define a struct Area that has two private variable members;

\* units of type string and area\_value of type float.

\*/

struct Area

{

private:

// Add your code here!

string unit;

float AreaValue;

public:

// Setter

void setAreaValue(float area\_value) {

AreaValue = area\_value;

}

// Getter

float getAreaValue() {

return AreaValue;

}

void setUnit(string units) {

unit = units;

}

string getUnit() {

return unit;

}

};

int main()

{

/\*

\* Modify program to create a dynamic variable of type Area.

\*/

// Add your code here!

Area \*area = new Area();

/\*

\* Input from the keyboard the area\_value and its units.

\*/

float value;

string unit;

cout << "Enter the value of the area: " << endl;

cin >>value;

cout << "Enter the units of the area: " << endl;

cin >> unit;

area->setAreaValue(value);

area->setUnit(unit);

// Add your code here!

/\*

\* Compute one-half and one-quarter of the area

\* and display the results

\*/

// Add your code here!

cout << "One half of the area : " << area->getAreaValue() / 2 << " " << area->getUnit() << endl;

cout << "One quarter of the area: " << area->getAreaValue() / 4 << " " << area->getUnit() << endl;

/\*

\* Destroy the dynamic variable at the end

\*/

// Add your code here!

delete area;

return 0;

}

Text

Description automatically generated

**Task 4**

To write a C++ program to perform matrix addition and subtraction. Matrices are basically stored in 2D arrays, which are created using dynamic memory allocation.

**Addition:** addition of two matrices is only possible when both matrices are of same dimensions. If this condition is true then corresponding elements are added.

**Requirements:**

* You are required to use dynamically allocated 1D arrays and treat them as 2D arrays i.e. to create an array of 2x3 you will get 6 compartments using new operator.
* You are supposed to write function of addition.
* Your main function should only contain function call to this function
* Your program will ask user about the dimensions of two matrices i.e. rows and columns. Then it will create these matrices dynamically and assign then random values and then perform addition of the two matrices.

**Code:**

#include<iostream>

#include<stdlib.h>

#include<time.h>

using namespace std;

void addMatrices() {

srand(time(NULL));

int row, col,row1,col1;

cout << "Enter the number of rows of matrix 1: " << endl;

cin >> row;

cout << "Enter the number of coloumns of matrix 1: " << endl;

cin >> col;

cout << "Enter the number of rows of matrix 2: " << endl;

cin >> row1;

cout << "Enter the number of coloumns of matrix 2: " << endl;

cin >> col1;

const int size=row\*col;

if (row == row1 && col == col1) {

//Addition algo of matrices

//Initializing the matrices according to the size

int\* matrix1 = new int[size];

int\* matrix2 = new int[size];

int\* addResult = new int[size];

//Assigning random values to matrices

for (int i = 0; i < size; i++) {

\*(matrix1 + i) = (rand() % 10);

}

for (int i = 0; i < size; i++) {

\*(matrix2 + i) = (rand() % 10);

}

cout << "The first Matrix is : " << endl;

for (int i = 0; i < row; i++) {

for (int j = 0; j < col; j++) {

cout << \*(matrix1+ i);

cout << " ";

}

cout << endl;

}

cout << "The second matrix is " << endl;

for (int i = 0; i < row; i++) {

for (int j = 0; j < col; j++) {

cout << \*(matrix2 +i);

cout << " ";

}

cout << endl;

}

for (int i = 0; i < size; i++) {

\*(addResult + i) = \*(matrix1 + i) +\*(matrix2 + i);

}

//displaying the result

cout << endl;

cout << "The Result of addition of two matrices is given as" << endl;

for (int i = 0; i < row; i++) {

for (int j = 0; j < col; j++) {

cout << \*(addResult+ i);

cout << " ";

}

cout << endl;

}

}

else {

//if the rows and coloumns are not same

cout << "The matrices should have the same order in order to add"<<endl;

}

}

int main() {

//Calling the add matrices

addMatrices();

}

**Output:**

**Text

Description automatically generated**