| ***Computer Engineering Department*** |
| --- |
| ***CE100L: Computing Fundamentals & Programming*** |

| ***Course Instructor: Usama Bin Shakeel*** | ***Dated: 24/01/2022*** |
| --- | --- |
| ***Teaching Assistant: Aqsa Khalid*** | ***Semester: Fall 2021*** |
| ***Lab Engineer: Nadir Abbas*** | ***Batch: BSCE2021*** |

# **Lab 14A. Dynamic Memory Allocation 1D**

| **Name** | **Roll number** | **Report**  **(out of 100)** | **Scaled to 10** | **Total**  **(out of 10)** |
| --- | --- | --- | --- | --- |
| NIMRA MAQBOOL | bsce21012 |  |  |  |

Checked on: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## **Objective**

The objective of this lab is to understand open ended problem solving.

## **Equipment and Component**

| **Component Description** | **Value** | **Quantity** |
| --- | --- | --- |
| Computer | Available in lab | 1 |

## **Conduct of Lab**

1. Students are required to perform this experiment individually.
2. In case the lab experiment is not understood, the students are advised to seek help from the course instructor, lab engineers, assigned teaching assistants (TA) and lab attendants.

## **Theory and Background**

A dynamic array is quite similar to a regular array, but its size is modifiable during program runtime. DynamArray elements occupy a contiguous block of memory. Once an array has been created, its size cannot be changed. However, a dynamic array is different. A dynamic array can expand its size even after it has been filled. During the creation of an array, it is allocated a predetermined amount of memory. This is not the case with a dynamic array as it grows its memory size by a certain factor when there is a need.

**Lab Task**

1. Write a C++ program that declares int variables x, y, z and int\* pointer variables p, q, r. Set x,

y, z to three different values. Set p, q, r to the addresses of x, y, z respectively.

(1) Print with labels the values of x, y, z, p, q, r, \*p, \*q, \*r.

(2) Print the message: Swapping pointers.

(3) Execute the swap code: r = p; p = q; q = r;

(4) Print with labels the values of x, y, z, p, q, r, \*p, \*q, \*r.

| void shittyPointers(){  cout<<"############################# TASK 1 ###########################";  int x=4; //declaring and giving it value of 4  int y=6; //declaring and giving it value of 6  int z=8; //declaring and giving it value of 8  int \*p;  int \*q; //initializing  int \*r;  p=&x;  q=&y; //giving path by reference  r=&z;  cout<<"the value of x = "<<x<<endl;  cout<<"the value of y = "<<y<<endl;  cout<<"the value of z = "<<z<<endl;  cout<<"the value of p = "<<p<<endl;  cout<<"the value of q = "<<q<<endl; //displaying  cout<<"the value of r = "<<r<<endl;  cout<<"the value of \*p = "<<\*p<<endl;  cout<<"the value of \*q = "<<\*q<<endl;  cout<<"the value of \*r = "<<\*r<<endl;  cout<<"############################# TASK 2 ###########################";  cout<<"swapping pointers ."; //printing statement  cout<<"############################# TASK 3 ##########################";  int \*num; //initializing  num=p; //storing value in num  p=q; //swapping  p=r;  r=num; //giving value of p in num  cout<<"the new address of p = "<<p<<endl;  cout<<"the new address of q = "<<q<<endl; //displaying address  cout<<"the new address of r = "<<r<<endl;  cout<<"############################ TASK 4 ###########################";  cout<<"the value of x = "<<x<<endl;  cout<<"the value of y = "<<y<<endl;  cout<<"the value of z = "<<z<<endl; //displaying values  cout<<"the value of p = "<<p<<endl;  cout<<"the value of q = "<<q<<endl;  cout<<"the value of r = "<<r<<endl;  cout<<"the value of \*p = "<<\*p<<endl;  cout<<"the value of \*q = "<<\*q<<endl;  cout<<"the value of \*r = "<<\*r<<endl;  }  OUTPUT: |
| --- |

2. Write a C++ program for 1D matrix addition and multiplication.

| ADDITION:  void addMatrix() {  int opt;  do {  cout << "1.For 1d row matrix addition." << endl;  cout << "2.for 1d column matrix addition." << endl; //displaying for option  cout << "3.exit." << endl;  cin >> opt;  switch (opt) {  case 1: {  rowAddition(); //calling  break;  }  case 2: {  columnAddition(); //calling  break;  }  case 3: {  cout << "you choose to exit..0\_0" << endl;  exit(1);  break;  }  default: {  cout << "you have enter invalid input" << endl;  }  }  } while (opt >= 1 && opt <= 3);  }  void rowAddition() {  int columns;  cout << "please enter the columns of matrix = "; //taking number of columns  cin >> columns;  int arr[1][40];  int arr1[1][40]; //initializing  int arr2[1][40];  cout << "please enter the values of matrix 1 = ";  for (int i = 0; i < 1; i++) {  for (int j = 0; j < columns; j++) {  cin >> arr1[i][j]; //getting values of matrix 1  }  }  cout << "please enter the values of matrix 2 = ";  for (int i = 0; i < 1; i++) {  for (int j = 0; j < columns; j++) {  cin >> arr2[i][j]; //getting values of matrix 2  }  }  for (int i = 0; i < 1; i++) {  for (int j = 0; j < columns; j++) {  arr[i][j] = arr1[i][j] + arr2[i][j]; //adding and then storing in array  }  }  cout << "the sum { ";  for (int i = 0; i < 1; i++) {  for (int j = 0; j < columns; j++) {  cout << arr[i][j] << " "; //displaying matrix  }  }  cout << "}";  cout << endl;  }  void columnAddition() {  int rows;  cout << "please enter thr number of rows in matrix = ";  cin >> rows;  int arrC[50][1];  int arrA[50][1];  int arrB[50][1];  cout << "please enter the values of matrix 1 = ";  for (int i = 0; i < rows; i++) {  for (int j = 0; j < 1; j++) {  cin >> arrA[i][j];  }  }  cout << "please enter the values of matrix 2 = ";  for (int i = 0; i < rows; i++) {  for (int j = 0; j < 1; j++) {  cin >> arrB[i][j];  }  }  for (int i = 0; i < rows; i++) {  for (int j = 0; j < 1; j++) {  arrC[i][j] = arrA[i][j] + arrB[i][j];  }  }  cout << "the sum \n";  for (int i = 0; i < rows; i++) {  for (int j = 0; j < 1; j++) {  cout << arrC[i][j];  }  cout << "\t\t" << endl;  }  cout << endl;  }  OUTPUT:    MULTIPLICATION:  void multiplyMatrix()  {  int row1;  int column1; //initializing  int row2;  int column2;  int sum=0;  cout<<"please enter the number of rows of matrix 1 = "; //getting rows of matrix 1 from user  cin>>row1;  cout<<"please enter the number of columns of matrix 1 = "; //getting columns of matrix 1 from user  cin>>column1;  cout<<"please enter the number of rows of matrix 2 = "; //getting rows of matrix 2 from user  cin>>row2;  cout<<"please enter the number of columns of matrix 2 = "; //getting columns of matrix 2 from user  cin>>column2;  int arrM[50][50];  int arrN[50][50];  int arrO[50][50]; //initializing arrays  if(row1==1 && column2==1){  if(column1==row2){  cout << "please enter the values of matrix 1 = ";  for (int i = 0; i < row1; i++) {  for (int j = 0; j < column1; j++) {  cin >> arrM[i][j]; //getting matrix 1 values  }  }  cout << "please enter the values of matrix 2 = ";  for (int i = 0; i < row2; i++) {  for (int j = 0; j < column2; j++) {  cin >> arrN[i][j]; //getting matrix 2 values  }  }  for(int i = 0; i < row1; i++)  for(int j = 0; j < column2;j++)  for(int k = 0; k < column1; k++) //multiplying and then adding the values to opt 1 by 1 matrix  {  arrO[i][j] =arrO[i][j] + arrM[i][k] \* arrN[k][j];  }  cout << endl << "multiplication is = " ;  for(int i = 0; i < row1; i++) {  for (int j = 0; j < column2; j++) {  cout << " " << arrO[i][j]; //displaying  }  cout << endl;  }  else{  cout<<"the row 2 is not equal to the column2 so this can not be multiplied.. -\_-"; //condition if row of matrix 1 is not equal to column of matrix 2  }  }  else{  cout<<"the row1 and column2 must be equal to 1 (for 1D)"<<endl; //condition  }  }  OUTPUT: |
| --- |

#### **Assessment Rubric for Lab**

**Method for assessment:**

Lab reports and instructor observation during lab sessions. Outcome assessed:

a. Ability to conduct experiments, as well as to analyze and interpret data (P) b. Ability to function on multi-disciplinary teams (A)

c. Ability to use the techniques, skills, and modern engineering tools necessary for engineering practice (P)

| Performance metric | Mapping (task no. and description) | | Max marks | Exceeds expectation | Meets expectation | Does not meet expectation | Obtained marks |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1. Realization of experiment (a) | 1 | Functionality | 40 | Executes without errors excellent user prompts, good use of symbols, spacing in output. Through testing has been completed (35-40) | Executes without errors, user prompts are understandable, minimum use of symbols or spacing in output. Some testing has been completed (20-34) | Does not execute due to syntax errors, runtime errors, user prompts are misleading or non-existent. No testing has been completed (0-19) |  |
| 2. Teamwork (b) | 1 | Group Performance | 5 | Actively engages and cooperates with other group member(s) in effective manner (4-5) | Cooperates with other group member(s) in a reasonable manner but conduct can be improved (2-3) | Distracts or discourages other group members from conducting the experiment (0-1) |  |
| 3. Conducting experiment (a, c) | 1 | On Spot Changes | 10 | Able to make changes (8-10) | Partially able to make changes (5-7) | Unable to make changes (0-4) |  |
| 2 | Viva | 10 | Answered all questions (8-10) | Few incorrect answers (5-7) | Unable to answer all questions (0-4) |  |
| 4. Laboratory safety and disciplinary rules (a) | 1 | Code commenting | 5 | Observes lab safety rules; handles the equipment and parts with care and adheres to the lab disciplinary guidelines aptly (4-5) | Generally observes safety rules and disciplinary guidelines with minor lapses (2-3) | Disregards lab safety and disciplinary rules (0-1) |  |
| 5. Data collection (c) | 1 | Code Structure | 5 | Excellent use of white space, creatively organized work, excellent use of variables and constants, correct identifiers for constants, No line-wrap (4-5) | Includes name, and assignment, white space makes the program fairly easy to read. Title, organized work, good use of variables (2-3) | Poor use of white space (indentation, blank lines) making code hard to read, disorganized and messy (0-1) |  |
| 6. Data analysis (a, c) | 1 | Algorithm | 20 | Solution is efficient, easy to understand, and maintain (15-20) | A logical solution that is easy to follow but it is not the most efficient (6-14) | A difficult and inefficient solution (0-5) |  |
| 7. Computer use (c) | 1 | Documentation | 5 | Timely documented (4-5) | Late documented (2-3) | Not documented (0-1) |  |
|  | Max Marks (total): | | 100 | Obtained Marks (total): | | |  |

Lab Engineer Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_