# Q 1: Vehicle Class – A class called Vehicle is required by a programmer who is writing software for a car dealer. An object of the class Vehicle will consist of a registration number, the make of the vehicle, the year of manufacture and the current financial value of the vehicle. The first three of these will need to be set only at the time an object is created. The current value will also be set at the time of creation, but may need to be changed during the vehicle's lifetime. You will need to write three files (Vehicle.h, Vehicle.cpp and Q1.cpp)

Your implemented class must fully provide the definitions of following class (interface) functions. Please also write down the test code to drive your class implementation. Please note that we will be running your code against our test code and any segmentation faults or incorrect result will result in loss of marks.

class Vehicle{ private:

// think about the private data members...

public:

// provide definitions of following functions...

Vehicle();// default constructor Vehicle(char \*str); Vehicle(double);

Vehicle(const Vehicle &);

//implement mutators for all private data members

//implement accessors for all private data members

//you have to implement the following functions

// think about the parameters required and return type of the following functions addVehicle();//adds a new vechile

ageOfVehicle();//returns age of vehicle on the basis of year passed getVehicleDetails();//return detail of vehicle getVehicleDetailsAtIndex();//return detail of vehicle stored at certain index isMatching();//returns true if passed vehicle matches with it returnByMake();//returns the array of vehicles of specific make returnByValue();//returns the array of vehicles of specific value returnByYear();//returns the array of vehicles of specific year vehicleSold();//deletes the specific vehicle once its sold

~Vehicle();

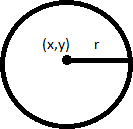
};

int main(){

/you to create N vehicles objects for the car dealer

}

# Q 2: Circle Class – A circle is identified by its center coordinates and its radius, as shown in the figure below. In the main file create N circles and implement a circle class whose specification is given below. You will need to write three files (Circle.h, Circle.cpp and Q2.cpp)



Your implemented class must fully provide the definitions of following class (interface) functions. Please also write down the test code to drive your class implementation. Please note that we will be running your code against our test code and any segmentation faults or incorrect result will result in loss of marks.

class Circle{ private:

// think about the private data members...

public:

// provide definitions of following functions... Circle();// default constructor

Circle(int x, int y);

Circle(int x, int y, int radius);

//you have to implement the following functions

// think about the parameters required and return type of the following functions setCenter();//set center of a circle

setRadius();//set radius of a circle getArea();//prints area of a circle

returnLargestCircle();//return the largest circle from the array of circles overlapping();//determines of two circles are overlapping or not overlappingCircles();//returns an array of circles overlapping the largest circle

~Circle();

};

int main(){

/you to create N circle objects and give user options to initialize them

}

# Q 3: Scheduler Class – In order to generate schedule of a project, key information is of tasks/activities to do in that project. You have to implement a class which takes the task information from the user and generate different information related to schedule of the project. You will need to write three files (Scheduler.h, Scheduler.cpp and Q3.cpp). The program should first ask the total number of tasks in the project.

Your implemented class must fully provide the definitions of following class (interface) functions. Please also write down the test code to drive your class implementation. Please note that we will be running your code against our test code and any segmentation faults or incorrect result will result in loss of marks.

# You will have to learn Critical Path Method here: https://[www.workamajig.com/blog/critical-path-](http://www.workamajig.com/blog/critical-path-) method

struct task{ int id; int dur;

int s\_Time; //start time of each task

int e\_Time; //end time of each task

int\* dep; /\*list of predecessors of this task - To simplify we assume that a higher number task will depend on a lower number task e.g. T2 can depend on T1 OR T4 can depend on T2 but the opposite is not true.\*/

};

class Scheduler{ private:

// think about the private data members...

public:

// provide definitions of following functions... Scheduler();// default constructor

Scheduler(task\* ts, int n);//initialized the project with n tasks

//you have to implement the following functions

// think about the parameters required and return type of the following functions

setTaskDuration();//change task duration of all tasks

set\_nth\_TaskDuration();//change duration of a specific task printTaskDependencyList();//print dependencies of a specific task completionTime();//print completion time of the project printCriticalTasks();/\*returns array of critical tasks and displays them – sum of their duration should be equal to project completion time\*/

~Scheduler();//destructor

};

# Q 4: Implementation of Integer Class – Your goal is to implement “Integer" class. You will need to write three files (Integer.h, Integer.cpp and Q4.cpp). Your implemented class must fully provide the definitions of following class (interface) functions. Please also write down the test code to drive your class implementation. Please note that we will be running your code against our test code and any segmentation faults or incorrect result will result in loss of marks.

class Integer{

// think about the private data members...

public:

//include all the necessary checks before performing the operations in the functions

Integer();// a default constructor Integer(int);// a parametrized constructor Integer(String); // a parametrized constructor void set(int);//set value

int get()const;//get value at (i,j)

int bitCount(); //Returns the number of one-bits in the 2's complement binary

int compareTo(Integer); //Compares two Integer objects numerically. double doubleValue(); //Returns the value of this Integer as a double. float floatValue(); //Returns the value of this Integer as a float.

Integer plus(const Integer&); //adds two Integers and return the result Integer minus(const Integer&); // subtracts two Integers and return the result

Integer multiple(const Integer&); //multiplies two Integers and return the result Integer divide(const Integer&); //divides two Integers and return the result static int numberOfLeadingZeros(int i); /\*Returns the number of zero bits preceding the highest-order ("leftmost") one-bit in the two's complement binary representation of the specified int value.\*/

static int numberOfTrailingZeros(int i); /\*Returns the number of zero bits following the lowest-order ("rightmost") one-bit in the two's complement binary representation of the specified int value.\*/

static String toBinaryString(int i); //Returns string representation of i static String toHexString(int i); //Returns string representation of i in base16 static String toOctString(int i); //Returns string representation of i in base 8

};

# Q 5: Implementation of String Class – Implementation of String Class Your goal is to implement a generic “String” class using cstring, i.e. you will need to write three files (string.h, string.cpp and Q5.cpp). Your implemented class must fully provide the definitions of following class (interface) functions. Please also write down the test code to drive your class implementation. Please note that we will be running your code against our test code and any segmentation faults or incorrect result will result in loss of marks.

class String{

private:

// think about the private data members...

public:

// provide definitions of following functions... String();// default constructor

String(char \*str);// initializes the string with constant cstring String(const String &);// copy constructor

String(int x);// initializes a string of pre-defined size char getAt(int i);// returns the character at index [x] void setAt(int i, char c);// set the character at index [x]

String substr(int pos, int len);// returns a substring of length len from ‘pos’ String substr(int pos);// returns substring from the given position to the end. void append(char a);// append a char at the end of string

void append(String str );// append a String at the end of string

void append(char \*str );// append a constant c string at the end of string

int length();// returns the length of string

char \* tocstring();// converts a String to c-string

void display();// displays the string ..

bool isEmpty();// returns true if string is empty..

void copy(const String&);// Copy one string to another ...

void copy(const char \*);// copy cstring to String...

int find(char);// returns the index of character being searched. bool equal(String);// should return true if both strings are same int stoi();// function for converting a string to integer.

˜String();// destructor...

};

# Q 6: Implementation of True Dynamic Array Class – Your goal is to implement a generic “TrueDynamicArray" class. You will need to write three files (TrueDynamicArray.h, TrueDynamicArray.cpp and Q6.cpp). Your implemented class must fully provide the definitions of following class (interface) functions. Please also write down the test code to drive your class implementation. Please note that we will be running your code against our test code and any segmentation faults or incorrect result will result in loss of marks.

struct node{ int value; node\* next;

};

class TrueDynamicArray{ private:

// think about the private data members...

public:

// provide definitions of following functions... TrueDynamicArray();// a default constructor

TrueDynamicArray (int size);// a parametrized constructor initializing an Array TrueDynamicArray (int \*arr, int size);// initializes the Array with an existing TrueDynamicArray (const TrueDynamicArray &);// copy constructor

int getAt(int i);// returns the integer at index [i]

void setAt(int i, int val);// set the value at index [i]

TrueDynamicArray subArr(int pos, int siz);// returns subArray of ‘siz’ from ‘pos’

TrueDynamicArray subArr(int pos);// returns a sub-Array from ‘pos’ to end

void push\_back(int a);// adds an element to the end of the array

int pop\_back();// removes and returns the last element of the array

int insert(int idx, int val);// inserts the value val at idx int erase(int idx, int val);// erases the value val at idx int length();// returns the size of the Array

void clear();//clears the contents of the Array

int value(int idx);//returns the value at idx

void assign(int idx, int val);//assigns the value val to the element at index idx

void display();// displays the Array

bool isEmpty();// returns true if the Array is empty

bool equal(TrueDynamicArray);// should return true if both Arrays are same int sort();// sorts the Array. Returns true if the array is already sorted void reverse():// reverses the contents of the array

˜TrueDynamicArray();

};

# Q 7: Implementation of Matrix Class – Your goal is to implement a generic “Matrix" class. You will need to write three files (matrix.h, matrix.cpp and Q7.cpp). Your implemented class must fully provide the definitions of following class (interface) functions. Please also write down the test code to drive your class implementation. Please note that we will be running your code against our test code and any segmentation faults or incorrect result will result in loss of marks.

class Matrix{

// think about the private data members...

// the matrix should store real numbers

public:

//include all the necessary checks before performing the operations in the functions

Matrix();// a default constructor

Matrix(int, int);// a parametrized constructor Matrix(const Matrix &);// copy constructor

void set(int i, int j, float val);//set value at (i,j)

float get(int i, int j)const;//get value at (i,j)

Matrix& assign(const Matrix &);//assigns (copies) a Matrix. Returns the same Matrix add(const Matrix &);//adds two Matrices and returns the result

Matrix subtract(const Matrix &);//subtracts two Matrices and returns the result Matrix multiply(const Matrix &);//multiplies two Matrices and returns the result Matrix multiplyElement(const Matrix &);//Elementwise multiplies two Matrices and returns the result

Matrix add(float );//assigns a constant to every element

Matrix multiply(float);//multiplies every element with a constant

void input(); // takes input in every element of matrix

void display(); // prints every element

˜Matrix();

};

# Q 8: LudoPlayer – Your goal is to implement a the Ludo game. You will need to write three files (LudoPlayer.h, LudoPlayer.cpp and Q8.cpp). When game begins ask how many players the user wants to simulate, assign different colors to each user or ask his/her preference. By default all tokens will be at position 0 and depending on the dice roll will move to next position. A token reaches the destination when it has crossed all positions on the board.

Your implemented class must fully provide the definitions of following class (interface) functions. Please also write down the test code to drive your class implementation. Please note that we will be running your code against our test code and any segmentation faults or incorrect result will result in loss of marks.

class LudoPlayer{

// think about the private data members...

// the matrix should store real numbers

public:

LudoPlayer();// a default constructor, to initialize position of each token LudoPlayer();/\* a parametrized constructor (color chosen by the player) – use integers for different colors\*/

//you have to implement the following functions

// think about the parameters required and return type of the following functions

movePlayer(); rollDice(); checkPosition(); displayActiveToken(); displayAllTokens();

˜LudoPlayer();

};

int main(){

/\*You need to simulate the Ludo game here. Ask for the number of players who want to play the game. After each turn display the position of each token of player playing the game\*/

}