**GREEDY CAR**

**A**

**Project Report**

Submitted in partial fulfillment of the

requirements for the award of the degree of

**BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE & ENGINEERING**

**School of Computer Science**

**Department of Systemics**

**UNIVERSITY OF PETROLEUM AND ENERGY STUDIES**

**Bidholi, Via Prem Nagar, Dehradun, Uttarakhand**

**Aug - Dec 2020**

**CANDIDATE’S DECLARATION**

We hereby certify that the project work entitled **“*Greedy Car*”**in partial fulfillment of the requirements for the award ofthe Degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING with specialization in **CYBER SECURITY AND FORENCICS** and submitted to the School of Computer Science, Department of Systemics, University of Petroleum & Energy Studies, Dehradun, is an authentic record of my/ our work carried out during a period from **Aug**-**2020** to **Dec-2020** under the supervision of **Mr. Vidyanand Mishra, Assistant Professor, Dept. of Systemics.**

The matter presented in this project has not been submitted by us for the award of any other degree of this or any other University.

|  |
| --- |
| Anish Bansal |
| R134218020 |

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date: \_\_\_\_\_\_\_\_\_\_2020

**Dr. Neelu Jyoti Ahuja**

HOD – Systemics

School of Computer Science

University of Petroleum & Energy Studies

Dehradun – 248 001 (Uttarakhand)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ACKNOWLEDGEMENT**

We wish to express our deep gratitude to our guide **Mr. Vidyanand Mishra**, for all advice, encouragement and constant support he has given us throughout our project work. This work would not have been possible without his support and valuable suggestions.

We sincerely thank to our respected Program Head of the Department, **Dr. Neelu Jyoti Ahuja**, for her great support in doing our project in **Human Resource Information System** at **SoCS**.

We are also grateful to **Dr. Manish Prateek,** Dean- SoCS, UPES for giving us the necessary facilities to carry out our project work successfully.

We would like to thank all our **friends** for their help and constructive criticism during our project work. Finally, we have no words to express our sincere gratitude to our **parents** who have shown us this world and for every support they have given us.

|  |  |
| --- | --- |
| Name | Anish Bansal |
| Roll No. | R134218020 |

**ABSTRACT**

The project aims at taking a step into the field of data structures and algorithms by creating a program which consists of a car traveling in such a way that it collects maximum number of coins which are placed on three different roads of equal length with each road having equal or unequal number of coins. When the program starts coins are randomly placed on these roads, each run of the program results in different distribution of coins rather than having fixed coin position. The car runs with certain constant velocity, move to left or right road and doesn’t accelerate or decelerate in the course of its run. It uses a certain kind of algorithm to collect maximum coins possible until it reaches the end of the track.

**INTRODUCTION**

The first step towards an understanding of why the study and knowledge of algorithms are so important is to define exactly what we mean by an algorithm. According to the popular algorithm’s textbook “Introduction to Algorithms” an algorithm is any well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values as output. In other words, algorithms are like road maps for accomplishing a given, well-defined task. On the other hand, we have data structures. Data Structure can be defined as the collection of data objects which provides a way of storing and managing data in the computer so that it can be used. Data Structures are widely used in almost every aspect of Computer Science for simple as well as complex computations. A right selection of data structure can enhance the efficiency of computer program or algorithm in a better way. Therefore, in our project we will use Algorithms and data structure to I 2 implement our program. Firstly, we will use 2-Dimensional array for creating three different roads on which our car runs and for randomly placing coins on these roads we will be using Linked List and rand () function. Further we will use searching algorithm for car to judge the path such that it collects maximum number of coins. Implementation of Linked List and 2D array for randomly placing coins on different positions on 3 different tracks is given below-:

1. For the representation of the three different tracks we will use 2D array. Each row of the 2D array will represent a track and total number of columns represents the size of each track.
2. We will execute “for” loop ten times. In each iteration “rand()” function will generate a random integer. The random integer generated by the rand() function must be in the range of [0,N) where N= total number of cells in the 2D array for example in the 2D array given below total number of cells is equal to 24.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 |  |  |  |  |  |  |  |  |
| 1 |  |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |  |

The rand () function returns a pseudo-random number in the range of 0 to RAND \_MAX. RAND\_MAX: is a constant whose default value may vary between implementations but it is granted to be at least 32767. So, to limit the range up to 23 we will us modulo (%) operator.

For Example: a=rand () %24.

1. The random numbers generated in the range of 0 to 23 will help us to select random nodes of the link list whose nodes are numbered 0 to 23, with each node also having address to one cell of the 2D array. For example, in the Linked List given below

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | Address of (0,0) of 2D array |  |  | 1 | Address of (1,0) of 2D array |  |  | 2 | Address of (2,0) of 2D array |  |  | 3 | Address of (0,1) of 2D array |  |  | 4 | Address of (1,1) of 2D array |  |

Node-0 of the Linked List has address of 0th row ,0th columns cell of the 2D array and likewise Node-1 has address of (1,0), Node-2 has address of (2,0), Node-3 has address (0,1), Node-4 has address (1,1) cell of the 2D array. Thus, coins will be placed on the address corresponding to the node selected of the 2D array.

After placing coins on the 3 tracks, we will use searching algorithm for judging path to be taken by the car.

In searching algorithm, we will search for coin in one column at time. The column having coin placed is the position of the car and likewise we will search in subsequent columns.

**PROBLEM STATEMENT**

Identifying a certain path such that car collects maximum number of coins.

**LITERATURE REVIEW**

|  |  |  |  |
| --- | --- | --- | --- |
| Title | Links | Authors | Remarks |
| Multidimensional Arrays | [1],[5] | Harsh Agarwal, Ansh Khurana, Rishabh Prabhu. | The base of this whole project is a 2D Array. And hence we got onto the internet and gathered as much information and knowledge about working with Multidimensional Arrays |
| Linked Lists | [2],[6],[7],[8],[10] | Ashwani Khemani, Rath Bhupendra, Ajay Kumar, Ankit Dalal, Ankur Atrr | To insert coins in any track we will have to make use of Linked Lists only. And hence we surfed the internet for any information about working with Linked Lists. |
| rand () Function | [3],[9] | Shivam Pradhan | To generate coins at random positions throughout the track, we need to use the rand () function that helps us to generate a random integer number. |
| Computer Graphics | [11] | - | The last step after creating the backend part of the project is to create an Interface for the car and the tracks. We will need the help of Computer Graphics for that. |

**OBJECTIVES**

To create a GUI based application consisting of a car traveling in such a way that it collects maximum number of coins.

Sub Objectives:

* To make graphical interfaces for roads and placing coins randomly on them.
* To make car identify the position of the coins on the roads.
* To show the car running on certain path such that user can identify the path taken.

**METHODOLOGY**

The only Programming language that will be used to make this project is C Programming Language. Since this is a purely C programming language-based project, we will also make use of all the concepts that we know of or can learn through the resources that are available to us. The basic functions and working have already been described but this section will describe the step-by-step implementation of this project. Below are the steps for the implantation: -

1. **Create a Multidimensional (2D) Array** – A 2D Array with 3 rows and N number of columns will be created to represent 3 tracks of N length. Only 1 car will traverse through any of the rows that fulfil our scenario/condition.
2. **Use Linked List to place coins inside the tracks** – A Linked List will be used to place coins inside positions of the tracks. Each tab will have an address associated to them which corresponds to an address inside the Multidimensional array.
3. **rand() function to generate coins at random positions** – Inside the Linked List we will use the rand() function to generate coins at random positions inside the multidimensional array.
4. **Use a searching Algorithm to determine the position of the coins** – To determine the position of the coins on the tracks, we will use Searching Algorithms column by column.
5. **Using Computer Graphics to create interfaces for the car, tracks and coins** – The last step in the coding process will be to design graphical interfaces for the car, the tracks and the coins. This will be the part that the user will be able to observe when the project is finally complete.

**Max. Coin Collection Algorithm:**

* 2-Dimensional array is used to make 3 separate tracks on which our car travels therefore each track has a size of n. So, there are 3 rows and n column in this 2dimesional array.
* For searching the coins, we search in every column one by one, but as our car can only switch to its left or right track during its run to collect coins and can only make one transition in every column. Therefore, car can only move to next or previous row of 2d array.
* As we are using searching in every column one by one therefore, we should skip collecting the coins which requires the car to make 2 transition of rows in a column. This is achieved by introducing the condition that the modulus of difference between next address car position and current address position should not be equal to 2n+1(if current address is greater than the previous address) and not equal to 2n-1(if previous address is greater than the current address.

**Code snip:**

for (int j=0; j<n; j++)

{

for (int i=0; i<3; i++)

{

if((arr+n\*i+j)>prev)

flag2=1;

else if((arr+n\*i+j) <prev)

flag2=0;

if(\*(arr+n\*i+j)!=0&&abs((arr+n\*i+j)-prev)!=(2\*n+1)&&flag2==1||\*(arr+n\*i+j)!=0&&abs((arr+n\*i+j)-prev)!=(2\*n-1)&&flag2==0||\*(arr+n\*i+j)!=0&&flag==0)

{

coinColl=coinColl+\*(arr+n\*i+j);

\*(arr+n\*i+j)=9;

prev=arr+n\*i+j;

i=3;

flag=1;

}

}

}

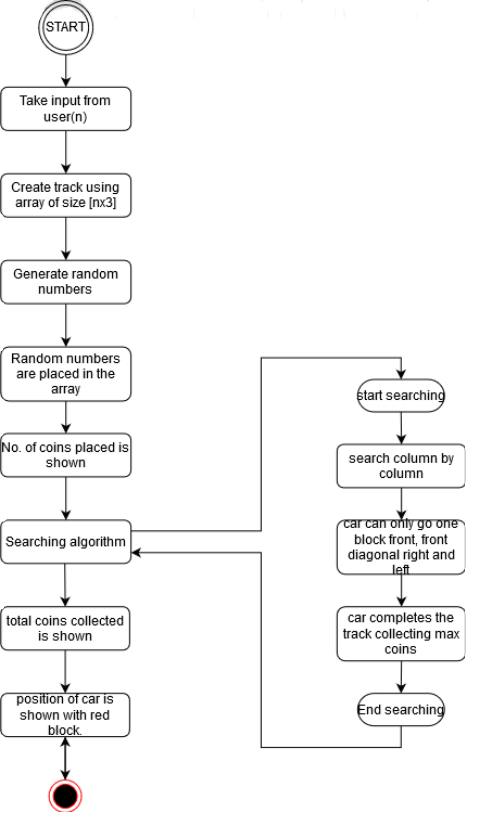
printf ("Coins COLLECTED= %d\n", coinColl);

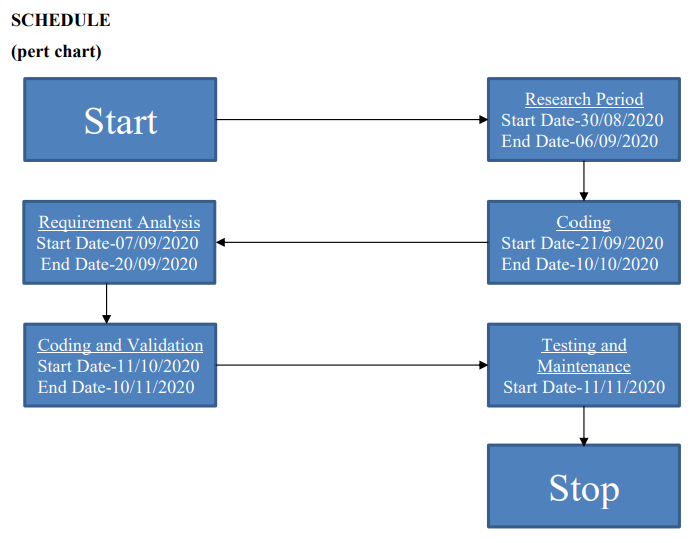
**Time complexity: O(3\*j)**

**Space complexity: O(3\*n) + O(3\*n)**

**LinkedList 2D Array**

**Flow Chart:**

****

****

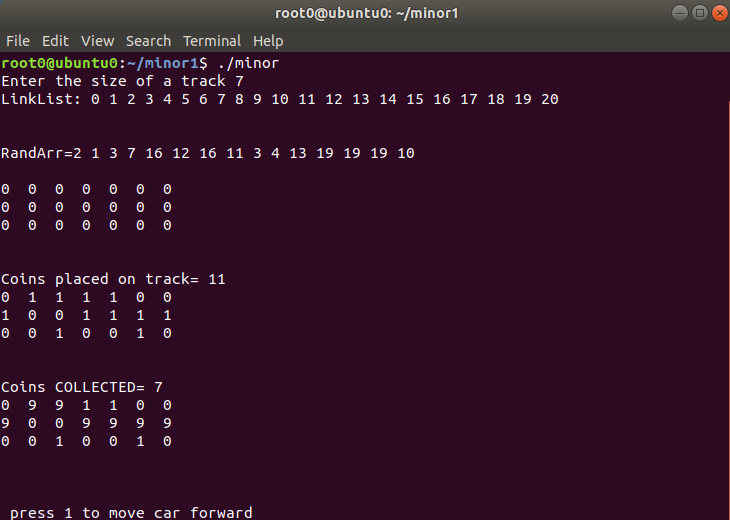
**OUR PROGRESS SO FAR:**

As of now, we have successfully implemented the backend part, that consists of:

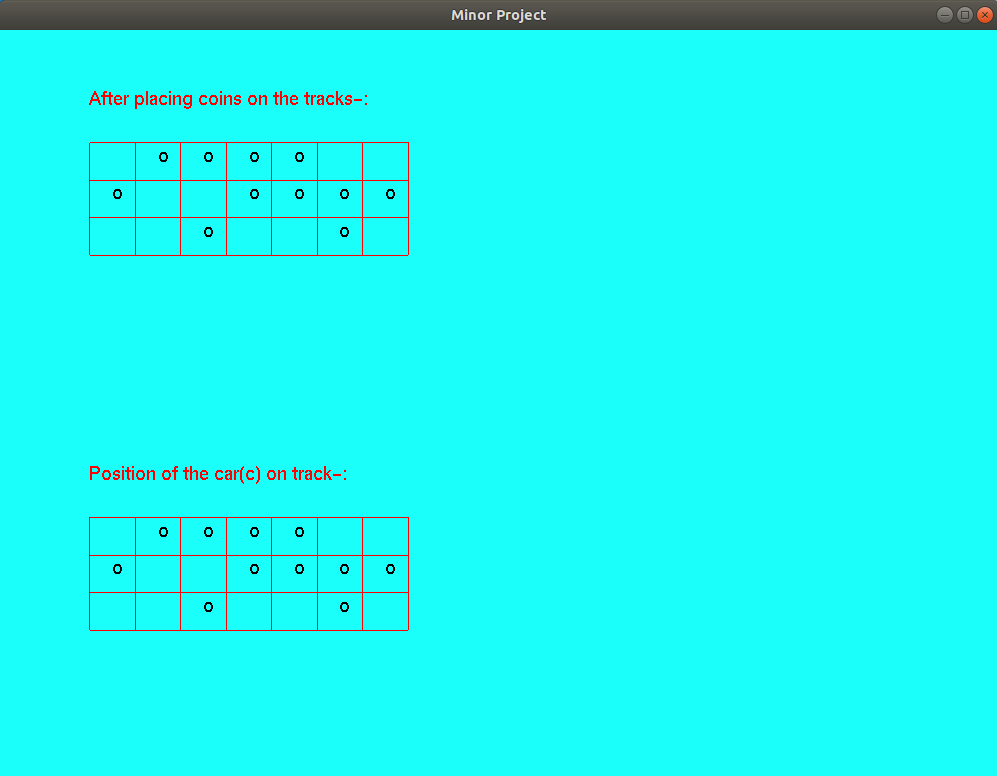
* Create a Multidimensional (2D) Array
* Use Linked List to place coins inside the tracks
* Create rand() function to generate coins at random positions
* Use a searching Algorithm to determine the position of the coins

Here is a sample output:

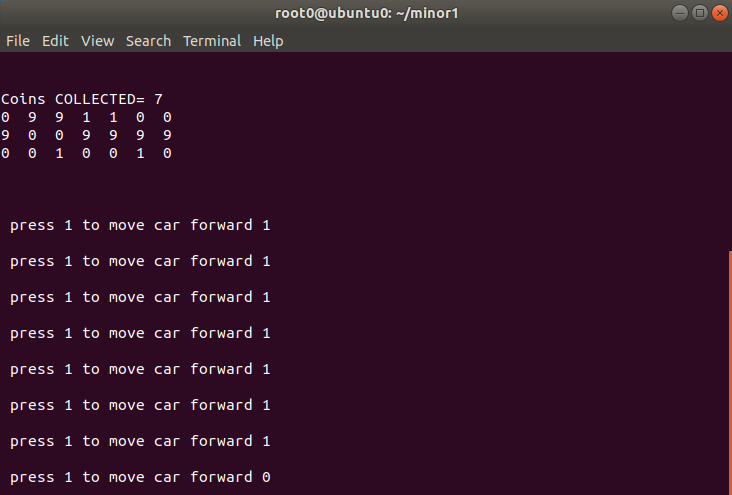
1. Size of track=7

****

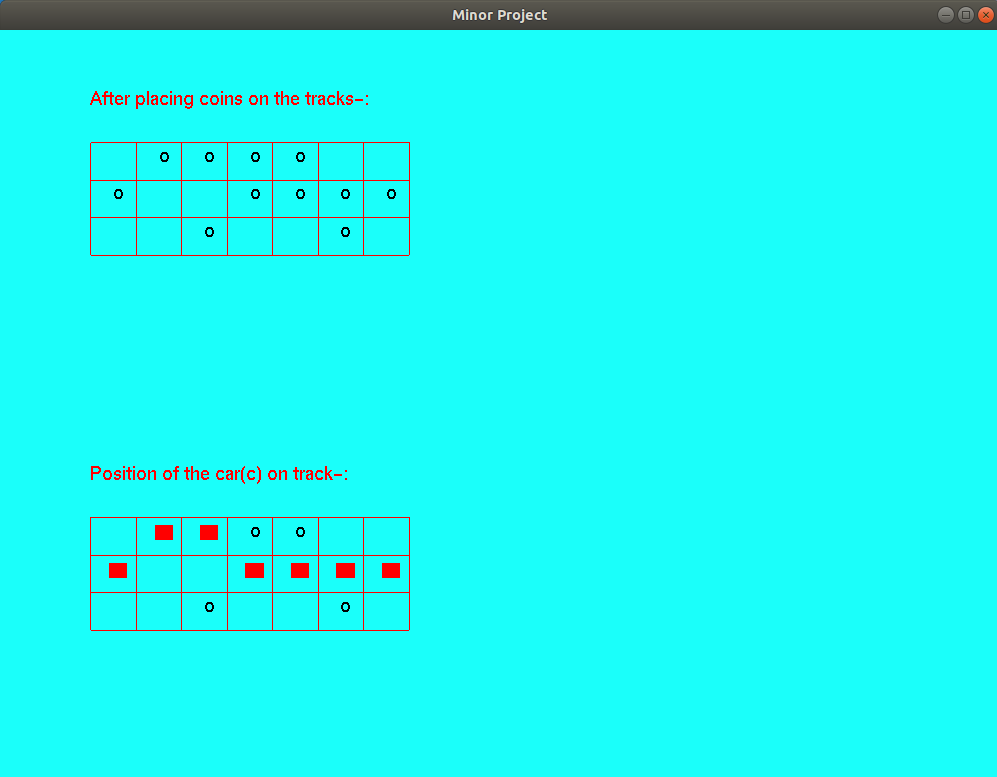
The openGL window will open as shown:



Now, to start the car we need to enter the number 1 in terminal window. Every time we enter 1 the car will move one block. To stop the car at the end of the track we can enter any number other than 1.



According to the inputs the car will move until it reaches the end of track while collecting the maximum number of coins as shown:



**Applications where this algorithm can be used:**

* 2D arrays, commonly known as, matrix, are used in image processing. Thus, our algorithm can be used in image processing.
* It can be used in speech processing, in which each speech signal is an array.
* 2D array are also used in representing the real-world data’s like the population of people, infant mortality rate, etc. Thus, our algorithm can search the required data easily from 2D array.
* Literally In every application of 2D array our algorithm can be used to search a required data from that array ex- Google map distance matrix, Image pixel matrix etc.
* It can be used in many games which requires our character to collect something in between the path and completes the race. Ex- need for speed, temple run, subway surfer, asphalt etc.

**REFERENCES:**

[1] <https://www.programiz.com/c-programming/c-multi-dimensional-arrays>

[Accessed: 25th Aug, 2020]

[2] <https://www.edureka.co/blog/linked-list-in-c/>

[Accessed: 25th Aug, 2020]

[3] <https://www.techonthenet.com/c_language/standard_library_functions/stdlib_h/rand.php>

[Accessed: 25th Aug, 2020]

[4] https://www.cs.cmu.edu/~ab/15-123S09/lectures/Lecture%2010%20- %20%20Linked%20List%20Operations.pdf

[Accessed: 27th Aug, 2020]

[5] <https://www.geeksforgeeks.org/multidimensional-arrays-c-cpp/>

[Accessed: 28th Aug, 2020]

[6] <https://www.geeksforgeeks.org/linked-list-set-1-introduction/>

[Accessed: 28th Aug,2020]

[7] <https://www.geeksforgeeks.org/linked-list-set-2-inserting-a-node/?ref=lbp>

[Accessed: 28th Aug, 2020]

[8] <https://www.geeksforgeeks.org/linked-list-set-3-deleting-node/?ref=lbp>

[Accessed: 28th Aug, 2020]

[9] <https://www.geeksforgeeks.org/rand-and-srand-in-ccpp/>

[Accessed: 29th Aug, 2020]

[10]<https://www.semanticscholar.org/paper/An-introduction-to-Linked-List-DalalAtri/d065c9c8bbd724d713e2b5075d85865b9905c835>

[Accessed: 29th Aug, 2020]

[11] <https://www.khronos.org/registry/OpenGL-Refpages/gl2.1/xhtml/>

[Accessed: 2nd Nov, 2020]