( **COMPUTATIONAL PHYSICS** )

**STUDY AND APPPLICATION OF MONTE CARLO METHOD**

**REPORT**

“ Submitted in partial fulfilment of the requirements for the course “

B.Sc (Physics)6th Semester

Under the guidance of

**Sir** **Soumyajit Pramanick**

****

Department Of Physics

St John College

Dimapur , Nagaland

797112

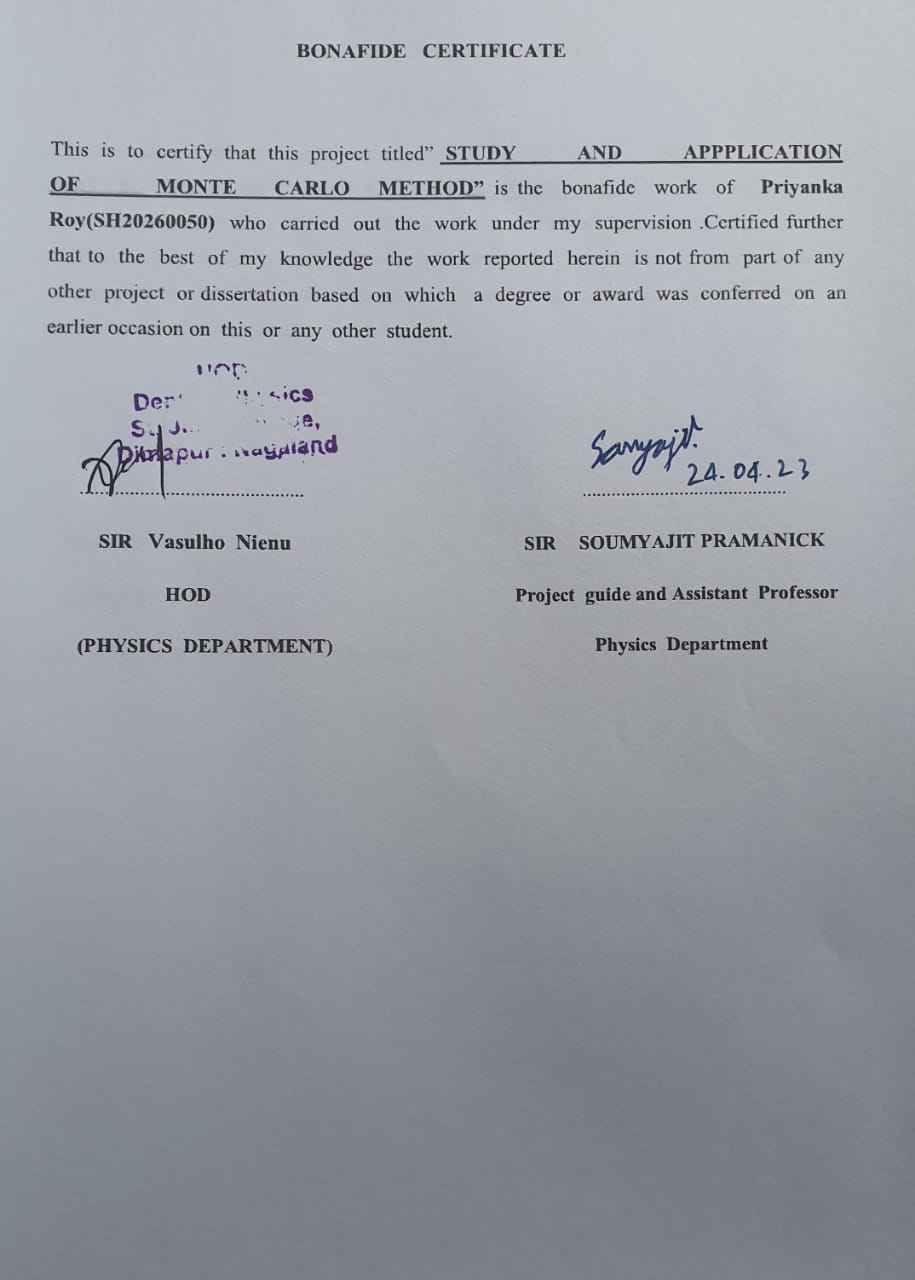
Submitted by

**Priyanka Roy**

(SH20260050)

B.Sc(Physics) 6th Semester

21st April 2023

****

**DECLARATION CERTIFICATE**

I ,hereby declare that the project work reprt entitled “**STUDY AND APPPLICATION OF MONTE CARLO METHOD”** ,which is being submitted to the St John College Nagaland for the award of the degree of B.Sc in PHYSICS is a bonafide report of the work carried out by me. The material contained in this project work report has not been submitted in any other University or Institution for the award of any degree.

Priyanka Roy

(SH20260050) Date:21st April 2023

B.Sc Physics,

St John College Nagaland

**ABSTRACT**

Computational physics involves solving physics problems through computational methods, ranging from simple to complex. One such method is the Monte Carlo method, which uses random number sampling to estimate the probability of an event and provide a numerical result that approximates the theoretical solution.

Several examples of Monte Carlo simulations, including the Monty Hall Problem, 1D and 2D random walks, and the Ising Model are presented in this work. These simulations are programmed in the C programming language to demonstrate that certain phenomena cannot be solved solely through theoretical methods. C is a beginner-friendly language commonly used in programming and the simulations were written using Integrated Development.The Ising Model is coded in python language for its better library functions.

Environments (IDEs) such as Visual Studio Code and Online C Compiler.

The Monty Hall Problem simulation that we implemented was designed to showcase the advantage of switching doors in order to increase the probability of winning. Through the simulation, we were able to reproduce the theoretical results that have been previously documented. By doing so, we gained a greater understanding of the counterintuitive nature of the problem and how the principles of probability play a crucial role in finding a solution.

In regards to the 1D and 2D random walks, we used simulations to investigate the statistical properties of particle movement. The simulations allowed us to delve into the behavior of random walks in varying dimensions and to gain a greater understanding of the nature of diffusion. We were able to uncover the underlying principles of diffusion and the impact of dimensionality on the phenomenon.

The Ising Model simulation was an essential aspect of our research, as it allowed us to study the behavior of ferromagnetic materials. We utilized a Metropolis algorithm to simulate the behavior of magnetic domains and to study the magnetic phase transition. The results that we obtained from the simulations were consistent with the theoretical predictions, providing us with a deeper insight into the fundamental principles that govern the behavior of ferromagnetic materials.

Overall, the Monte Carlo simulations of the Monty Hall Problem, 1D and 2D random walks, and the Ising Model were invaluable tools that provided us with a wealth of knowledge regarding the behavior of complex systems in physics. The algorithms that we developed in this study have significant potential for future applications in a wide range of physics problems and have the ability to pave the way for the development of new computational methods for tackling challenging problems in the field.

KEY WORDS: Monte Carlo Method,Stimulation,Probability,Random Numbers, Metropolis algorithm.

**AKNOWLEDGEMENT**

I am fortunate of doing this project on computational physics as computers have excited me since childhood days .This project have made me more acquainted of doing more upcoming projects with combination of Physics and Computer Science.

I, pay gratitude to God for keeping me in a good physical and mental health so that I could concentrate on this research ,without his will it would have not been possible. I have boundless respect for my guide **Sir** **Soumyajit Pramanick** for his efforts, patience, encouragement and guidance in throughout the completion of this research.

My heartfelt thanks to the Sir Vasulho Nienu (Assistant Professor ),Miss Rebecca Thapa (Assistant Professor) and Sir Lal Omega Boro (Research Scholar)whose comments and feedbacks were encouraging which helped to do the work more earnestly .

My humblest honor to the teaching staffs of Physics Department and Computer Science department who have been involved directly or indirectly in the completion of this work

**CONTENTS:**

|  |  |  |
| --- | --- | --- |
| CHAPTER N0 | TITLE | PAGES |
| 1. | INTRODUCTION | 8-11 |
| 2. | LITERATURE REVIEW | 12-14 |
| 3. | METHODOLOGY | 15-44 |
| 4. | DISCUSSION AND RESULTS | 45-49 |
| 5. | CONCLUSION | 50-52 |
| 6. | CODE AVAILABILITY | 53 |
| 7. | REFERENCES | 54-59 |

**I.INTRODUCTION.**

Computational Physics deals with finding the solution of physics questions ranging from simple to complex by computing.

Monte Carlo method is a method used to predict the probability of an event. It uses sampling of random numbers to give a numerical result. John von Neumann invented this method during the period of WWII for their decision-making purposes as the result given by this method reduces the decrease in chances of failing while taking risks.

It is widely used in the field of physics, engineering, and finance to solve problems that are difficult or impossible to solve analytically. It is named after the famous casino in Monaco, where games of predictions like roulette and craps are played.

Monte Carlo simulates a system's or process's behaviour using random integers. Let's say we want to determine the size of an oddly shaped garden in our property. We could apply the Monte Carlo method by dispersing numerous points inside the garden at random and counting how many of those points land inside the garden's perimeter. We can calculate the area of the garden by dividing the number of points inside the garden by the overall number of points. Our approximation becomes more precise the more points we scatter.

We need to create random numbers that are evenly distributed between 0 and 1 in order to apply the Monte Carlo technique. To create these random figures, we can either use a computer programme or physically random processes, like the radioactive element's decay.

The Monte Carlo method's ability to manage issues with numerous variables and intricate relationships between them is one of its advantages. Consider the scenario where we want to simulate the spread of a disease within a community. We could produce a wide range of potential outcomes based on various probabilities of infection, recovery, and death using the Monte Carlo technique. We can estimate the disease's overall behaviour and predict how it will impact the community by averaging across all the possible outcomes. However, the Monte Carlo method can be computationally expensive because it requires simulating many scenarios. Therefore, it is important to use efficient algorithms and parallel computing when possible.

The Monte Carlo method is a potent computational method that can be applied to resolve issues involving ambiguity and randomness. We can anticipate the behaviour of complex systems and estimate their behaviour by simulating a wide range of potential scenarios using random numbers.

Here the results are not exactly the same as the theoretical but an approximation of it.

Here ,some examples of Monte Carlo method are:

i. Monty Hall Problem.

ii. Random walk in 1D.

iii. Random walk in 2D.

iv. Ising Model.

The above examples of Monte Carlo method are computed in a programming language to prove that some phenomenon can’t only be solved theoretically.

I have used C programming language to write the codes for the examples. C language is the first basic language that we learn in programming world. It is a beginner’s friendly language.

Integrated Development Environments(IDEs): i)Visual Studio Code and ii)Online C Compiler

Some applications of Monte Carlo Method are:

i)It is applicable in stock investment to reduce risk for losing money when the shares goes up and down.

ii)Project management System.

iii)Monty Hall problems ie, the switching of choices.

iv)Business.

v)Artificial Intelligence.

**II. LITERATURE REVIEW**

The Monte Carlo method is a computational technique used to solve complex problems by simulating many random events or scenarios.This method is widely used in various fields such as finance, physics, engineering, and biology to model and analyze systems that are too difficult or expensive to study experimentally. The Monte Carlo method was first introduced in the 1940s by Stanislaw Ulam and John von Neumann and has since become a fundamental tool in many areas of research and industry[13].

The principles of the Monte Carlo method involve several key concepts that are critical to its successful application[11][12].The first principle is random sampling, which involves generating many random samples or scenarios, each of which represents a potential outcome of the system being modeled. The samples are generated using a random number generator, which produces a sequence of numbers with no discernible pattern. The sequence is typically uniform, meaning that each number has an equal probability of being generated[10][14][15].

.

The second principle is probabilistic modeling, which involves defining a set of probability distributions that describe the behavior of the system under different conditions. Probability distributions can be used to model a wide range of variables, including prices, temperatures, and velocities. The distributions can be continuous or discrete and can be defined using a wide range of statistical techniques. This principle is critical to ensure that the simulation accurately reflects the behavior of the system being modeled.

The third principle is simulation, which involves running the model for each of the random samples generated and recording the results. The simulation can be run using a computer program, which applies the probabilistic model to each of the random samples and generates a result for each sample. The simulation process is typically iterative and may involve refining the model or adjusting the simulation parameters to improve the accuracy of the results.

The fourth principle is statistical analysis, which involves using statistical techniques to extract meaningful information from the simulation results. This may involve calculating statistics such as means, variances, and probabilities to understand the behavior of the system under different conditions. The statistical analysis can be used to identify patterns, trends, and outwears in the simulation results.

The fifth principle is iteration, which involves repeating the simulation process to refine the results. This may involve adjusting the model parameters or generating more random samples to improve the accuracy of the results. The iteration process is critical to ensure that the simulation accurately reflects the behavior of the system being modeled.

The final principle is convergence, which refers to the point at which the simulation results converge to the true behavior of the system. The accuracy of the Monte Carlo simulation depends on the number of random samples generated, and as the number of samples increases, the simulation results converge to the true behavior of the system. Convergence can be monitored by calculating statistics such as the standard deviation or the coefficient of variation[10][14][15].

The Monte Carlo method has many applications in various fields. In finance, the Monte Carlo method can be used to simulate stock prices and estimate the value of financial derivatives. The method is used to model the uncertainty and risk associated with financial instruments and can be used to inform investment decisions. In physics, Monte Carlo simulations can be used to model complex systems such as molecular dynamics or nuclear reactors.

The method is used to simulate the behavior of particles, atoms, and molecules and can be used to design new materials and technologies. In engineering, the Monte Carlo method can be used to evaluate the safety of structures under extreme conditions. The method is used to simulate the behavior of structures under different loads and can be used to optimize the design of structures.

In biology, the Monte Carlo method can be used to simulate the behavior of biological systems such as protein folding or the spread of infectious diseases. The method is used to simulate the behavior of biological systems at the molecular, cellular, and population levels and can be used to develop new treatments and therapies[17].

**III. METHODOLOGY**

The Monte Carlo method is a computational technique that involves simulating a large number of random events or scenarios to solve complex problems. This method relies on random sampling, probabilistic modeling, simulation, statistical analysis, iteration, and convergence. The Monte Carlo method is widely used in various fields, including finance,physics, engineering, and biology.

When it comes to implementing Monte Carlo simulations, a C compiler can be used to generate optimized code, which can speed up the simulation process and yield more accurate results. The use of a C compiler can lead to more efficient code execution and reduced computation time, which is particularly useful for running large and complex simulations. The optimized code generated by a C compiler can help to optimize the simulation for specific hardware configurations, leading to improved simulation performance.

Overall, the combination of the Monte Carlo method and a C compiler can be a powerful tool for solving complex problems and generating accurate simulations. The efficient execution of code using a C compiler can reduce computation time and improve the accuracy of the simulation results. With its many benefits, the use of a C compiler in Monte Carlo simulations is becoming increasingly common in the fields that rely on the Monte Carlo method for solving complex problems.

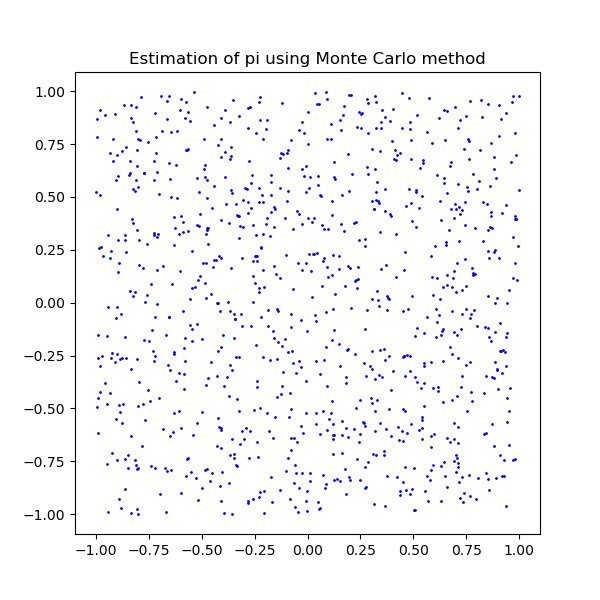
1st Example: Monte Carlo method.

The algorithm is designed to estimate the value of pi using Monte Carlo simulation techniques. It generates a pair of random numbers, x and y, each between 0 and 1, which correspond to a coordinate within a unit square. The algorithm then calculates the value of x²+y², which represents the square of the distance from the origin to the generated coordinate. This value is compared with 1, which corresponds to the radius of the unit circle. If the generated point falls within the unit circle, which means that the distance from the origin is less than 1, the algorithm increments the value of n.

The algorithm repeats this process for a large number of iterations, generating a sequence of points within the unit square and counting the number of points that fall within the unit circle. By dividing the number of points that fall within the unit circle by the total number of generated points, the algorithm estimates the ratio of the areas of the unit circle and the unit square. This ratio is equal to pi/4, which means that multiplying the ratio by 4 gives an estimate of pi.

This method of estimating pi using Monte Carlo simulation is a powerful technique that has found numerous applications in various fields such as computer science, physics, and engineering. The algorithm's simplicity and efficiency make it a popular choice for solving complex problems that are difficult to solve using traditional analytical methods.

Aim :Here you have an analytical value of pi ie,3.14 .We generate some random numbers which will range from 0 to 1 and by the formulas get an experimented value of pi which will also calculate the error and also the error percentage.

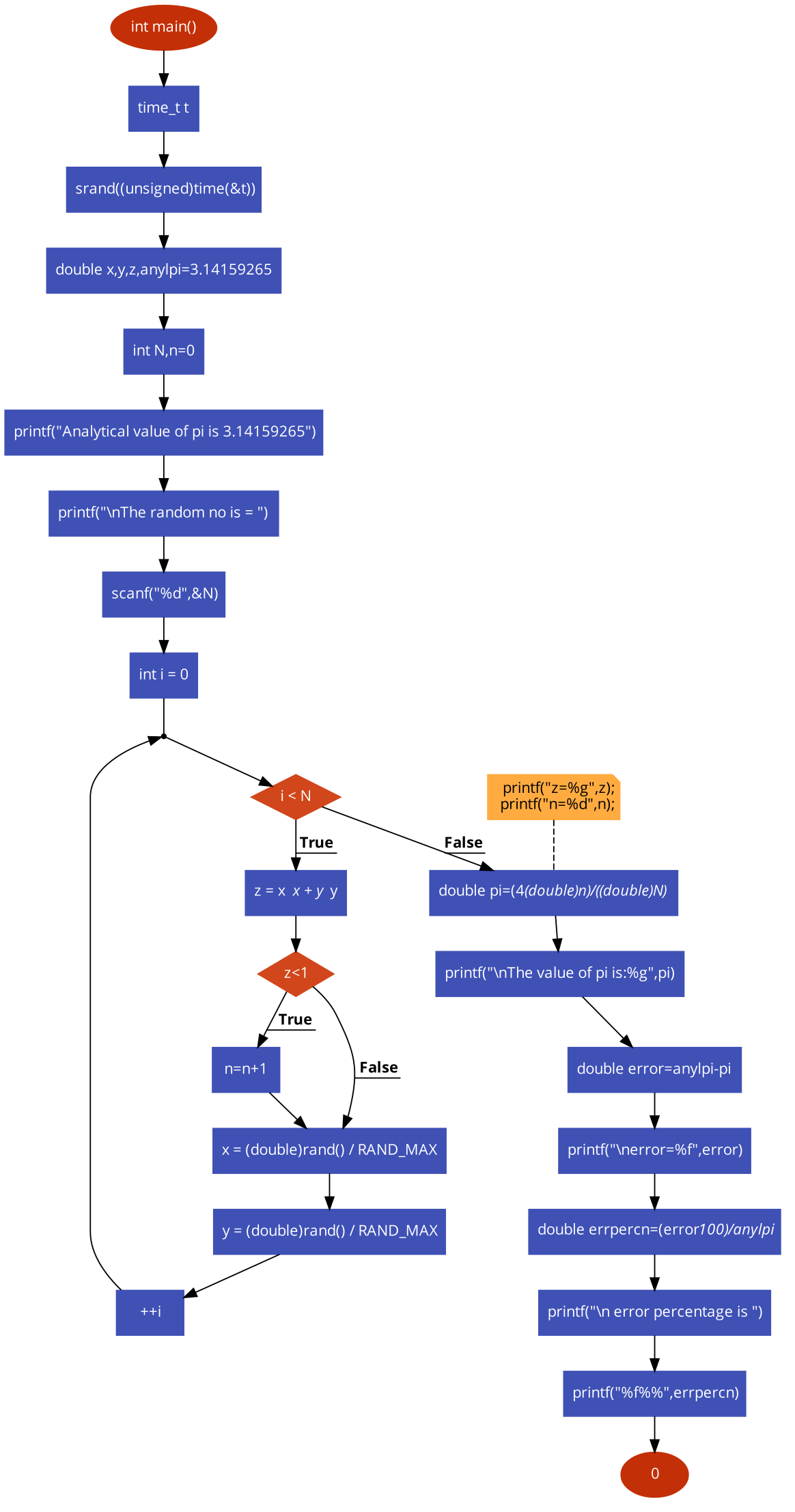


Algorithm/Pseudocode

1. Initialize variables t, x, y, z, anylpi, N, n to 0
2. Get the current time using the time function and assign it to t
3. Seed the random number generator with t using srand function
4. Print the value of analytical value of pi
5. Read the value of N from the user
6. Loop i from 0 to N-1:
7. Generate two random numbers x and y between 0 and 1 using rand function
8. Calculate z = x^2 + y^2
9. . If z < 1, increment the value of n by 1
10. Calculate the value of pi = 4 \* n / N
11. Calculate the error between the calculated value of pi and the analytical value of pi using the formula: error = anylpi - pi
12. Calculate the error percentage using the formula: errpercn = (error \* 100) / anylpi
13. Print the value of pi, error, and error percentage\

|  |  |  |  |
| --- | --- | --- | --- |
| ITERATION | ANALYTICAL VALUE OF PI | VALUE OF PI | ERROR % |
| 10 | 3.14159265 | 2.800000 | 10.864491% |
| 100 | 3.14159265 | 3.080000 | 1.954347% |
| 1000 | 3.14159265 | 3.139200 | 0.076144% |
| 10000 | 3.14159265 | 3.140800 | 0.025129% |
| 100000 | 3.14159265 | 3.142360 | 0.024399% |
| 1000000 | 3.14159265 | 3.141668 | 0.002402% |

FLOWCHART



2nd Example : Monty Hall Problem.

Suppose you're on a game show, and you're given the choice of three doors: Behind one door is a car; behind the others, goats. You pick a door, say No. 1, and the host, who knows what's behind the doors, opens another door, say No. 3, which has a goat. He then says to you, "Do you want to pick door No. 2?" Is it to your advantage to switch your choice?’[18]

Aim:To find the probability chances of winning and losing by switching the doors.

Here we are given choices in three doors ie,1,2,3 .There are trials ,opening door and closing door The percentage chances of winning and losing . The variable opening door and closing door generates some random numbers. You can enter the number of trials according to you.

Algorithm/Pseudocode

1. Start the program.
2. Initialize variables n, lost, and win to 0.
3. Read the value of n from the user.
4. Seed the random number generator with the current time using the srand function.
5. Loop i from 0 to n-1:
6. Generate two random numbers a and b between 1 and 3 using the rand function.

b. If b is equal to a, increment the value of win by 1.

c. Otherwise, increment the value of lost by 1.

6.Print the results of the experiment:

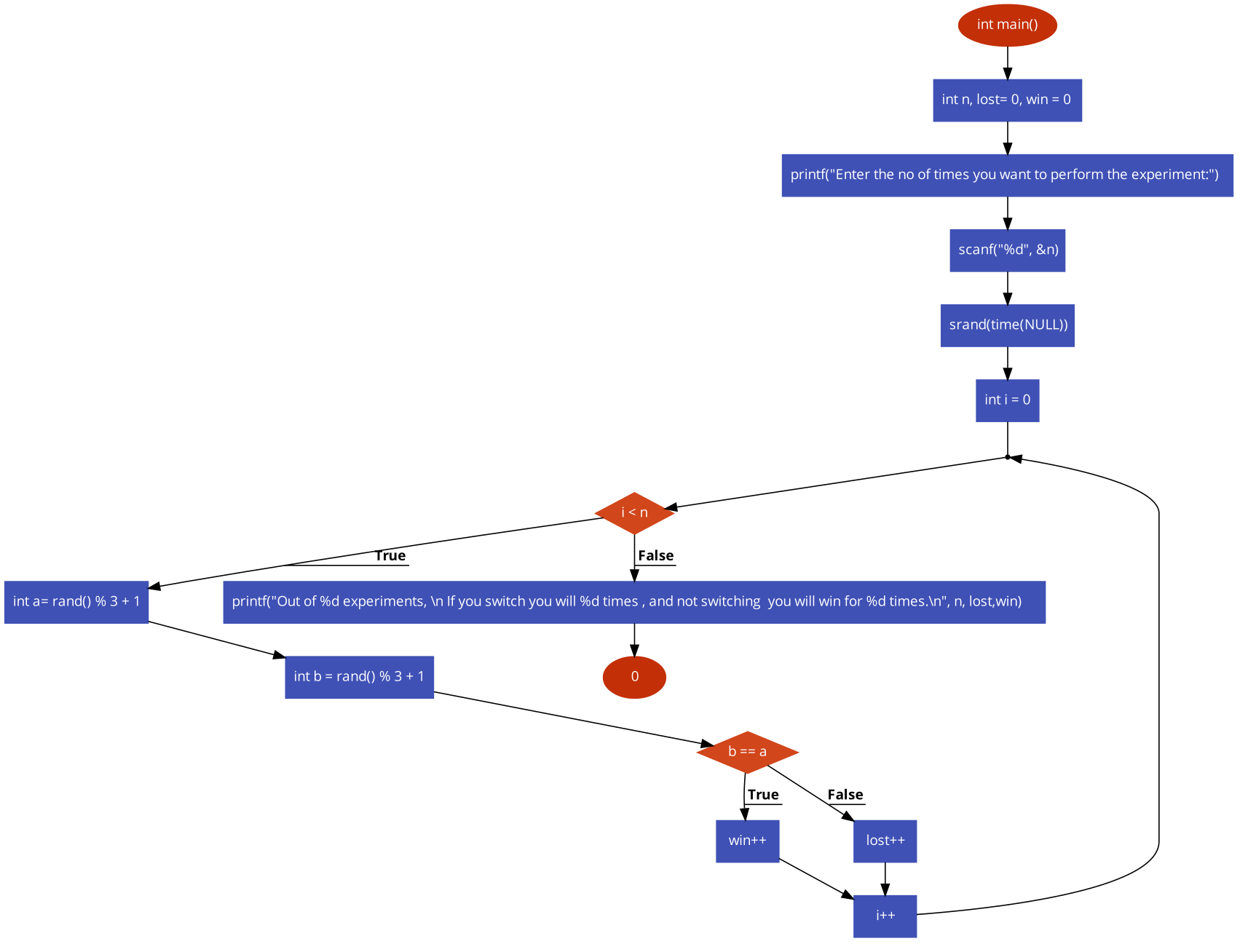
a.The number of times the experiment was performed.

b. The number of times the player would win if they switch.

c. The number of times the player would win if they don't switch.

7.End the program.

FLOWCHART



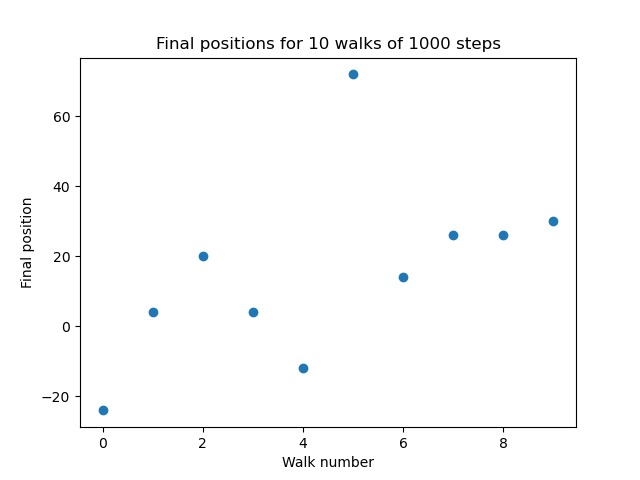
3rd Example: Random Walk in 1D

Aim: To find the average distance from origin and squareroot of average of average of distance covered if a walker takes random walk in 1D

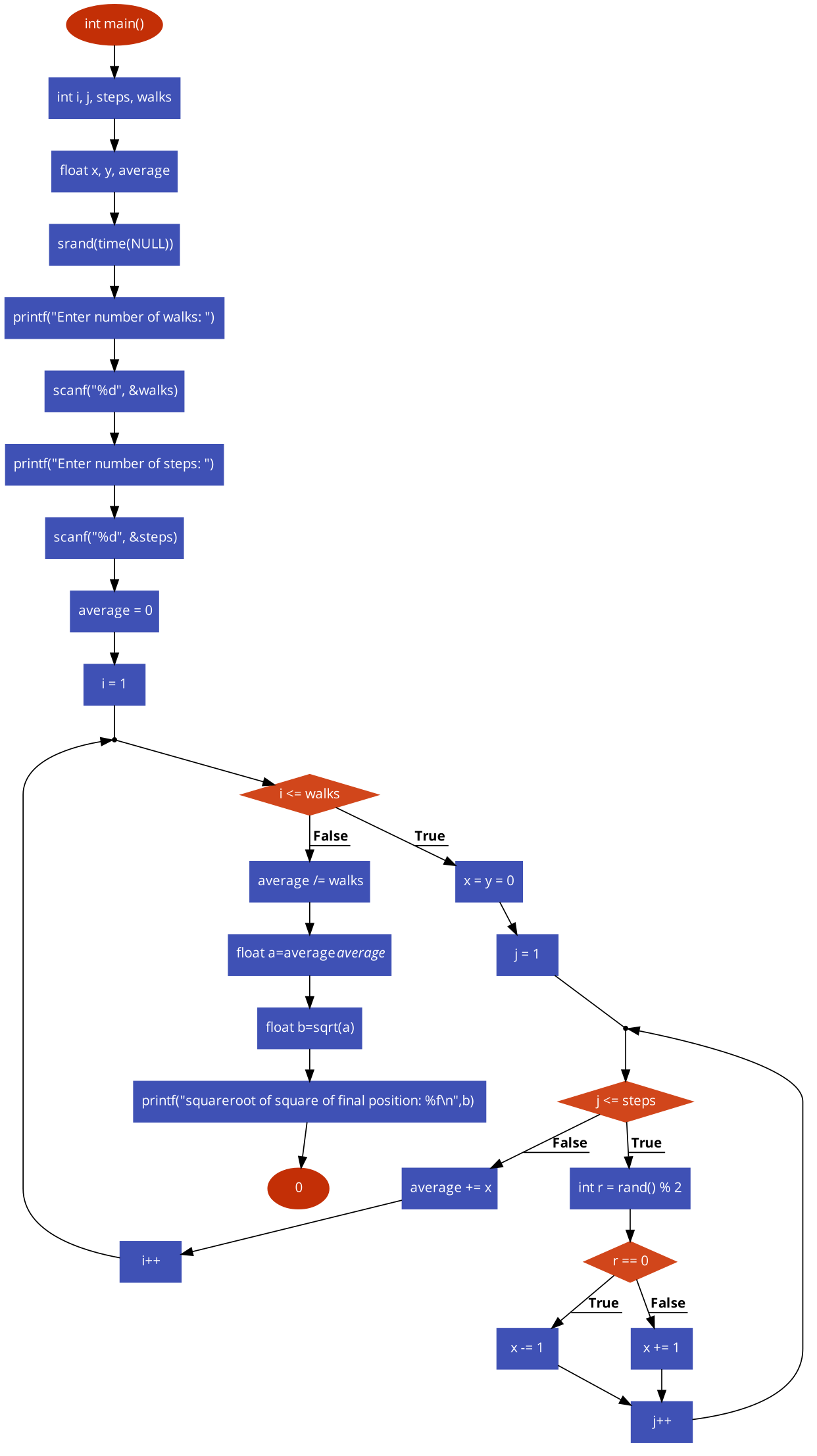
A walker starts from x=0 and in each step,goes in the positive or negative x direction with equal probability ,doing a 10 number of walks covering 3 steps total in 1 walk.

Algorithm/Pseudocode

1. Start the program.
2. Initialize variables i, j, steps, walks, x, y, average, a, and b to 0.
3. Read the value of walks from the user.
4. Read the value of steps from the user.
5. Seed the random numbeSr generator with the current time using the srand function.
6. Initialize the average variable to 0.
7. Loop i from 1 to walks: a. Set x and y to 0 b. Loop j from 1 to steps:
   1. Generate a random number r between 0 and 1 using the rand function.
   2. If r is 0, decrement the value of x by 1.
   3. If r is 1, increment the value of x by 1 . Add the value of x to the average variable
8. Divide the average variable by the number of walks to get the average position.
9. Calculate the square of the average position and store it in the variable a.
10. Calculate the square root of a and store it in the variable b.
11. Print the square root of the square of the final position.
12. End the program.



FLOWCHART



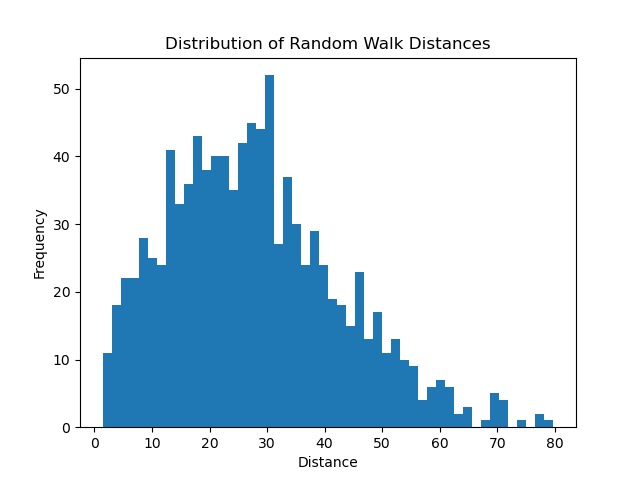
4th Example: Random Walk in 2D

Aim: To find the average distance from origin and root mean square( Rms) distance covered if a walker takes random walk in 2D

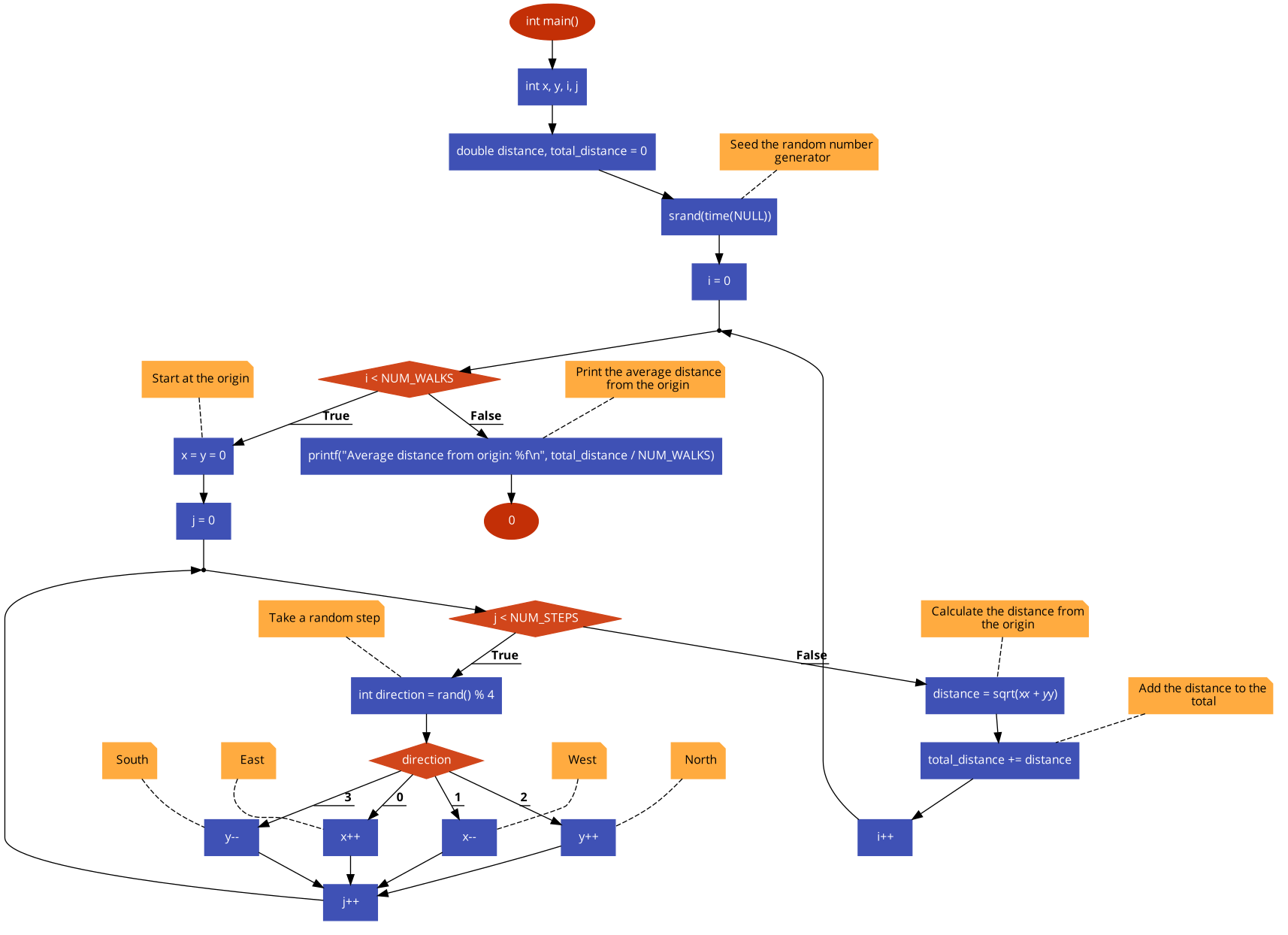
A walker starts from (0,0) and takes steps in east/west/north /south,with equal probability here the number of walks is 100 and steps covered in 1 walk is 1000

Algorithm/Pseudocode

1. Import necessary libraries: stdio.h, stdlib.h, time.h, math.h
2. Define the number of steps and walks using preprocessor macros: NUM\_STEPS and NUM\_WALKS respectively.
3. Declare integer variables x, y, i, and j and a double variable distance and total\_distance.
4. Seed the random number generator using srand function and pass time(NULL) as an argument.
5. Run a loop for NUM\_WALKS times to simulate the random walks.
6. Set the starting position to (0, 0) i.e. x and y to 0.
7. Run another loop for NUM\_STEPS times to take random steps.
8. Generate a random number between 0 and 3 inclusive using rand function to get a direction to move.
9. Depending on the direction obtained from the random number, update the values of x and y accordingly.
10. After completing the NUM\_STEPS, calculate the distance of the final position from the origin using the Pythagorean theorem and assign it to the variable distance.
11. Add the distance to the variable total\_distance to calculate the total distance from the origin for all the random walks.
12. After completing the NUM\_WALKS, calculate the average distance from the origin by dividing the total\_distance by NUM\_WALKS and print it using printf function.
13. End the program.



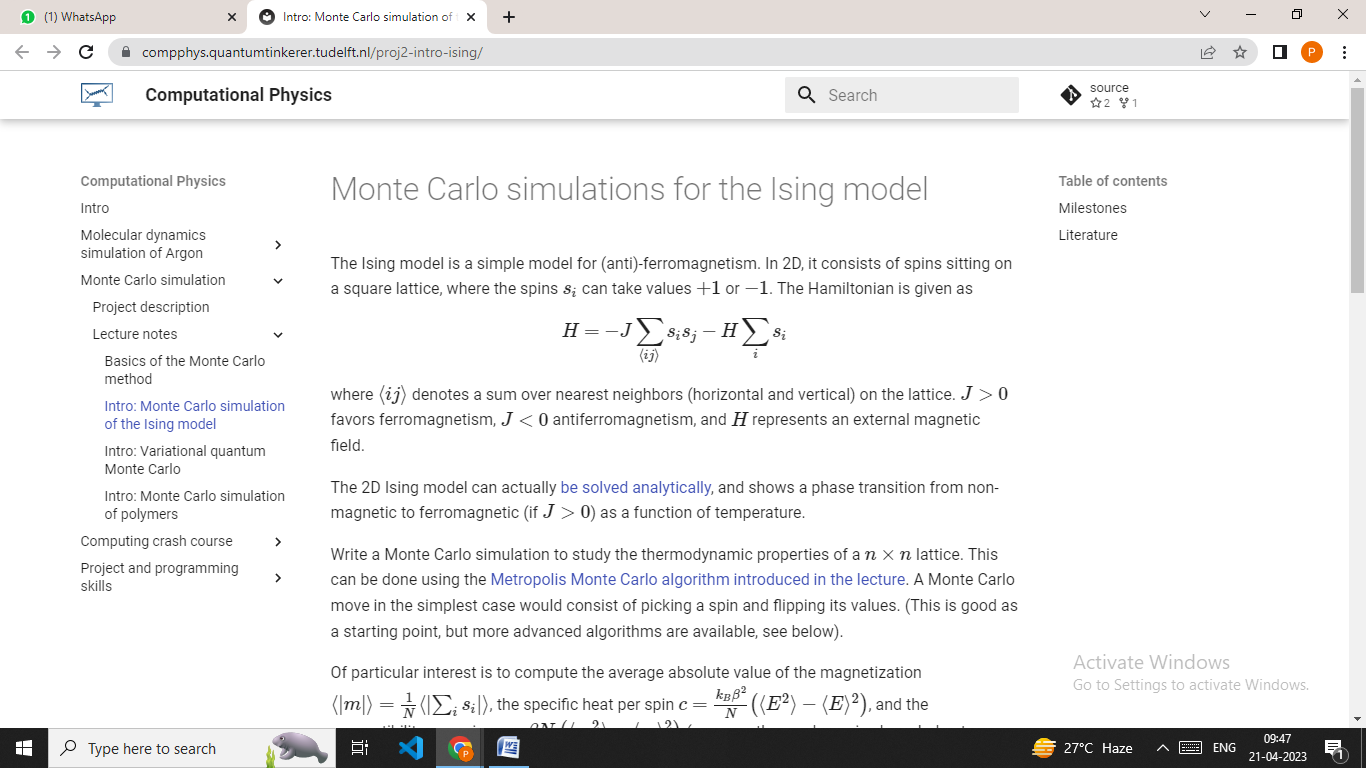
FLOWCHART



5th example: Ising Model in 2D

The Ising model is a statistical physics model used to study ferromagnetic materials in a two-dimensional lattice of discrete spins. It employs an energy function to govern the spins' behavior, and can be examined through Monte Carlo simulation techniques to observe phase transitions and other fascinating phenomena.

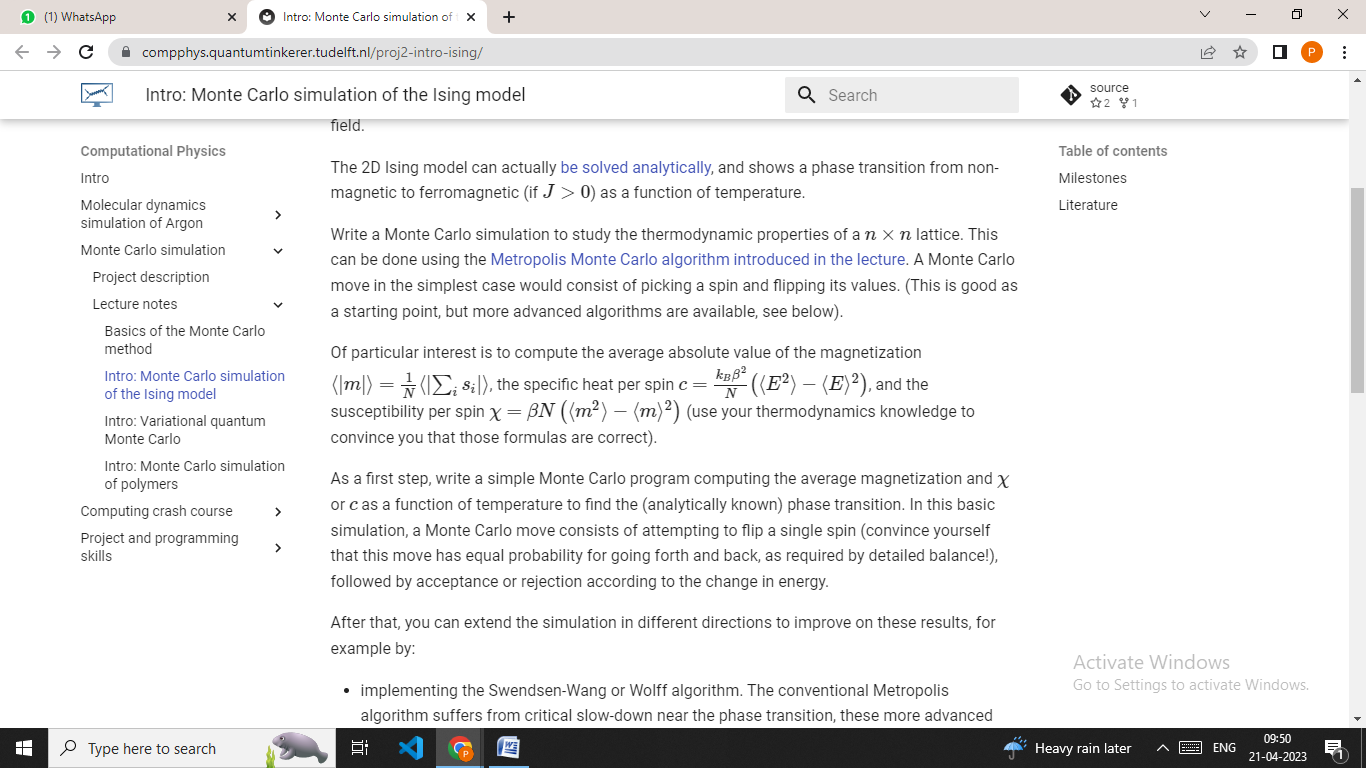
The Ising model is a mathematical model used to study (anti)-ferromagnetism. In 2D, this model is composed of spins located on a square lattice where the spins si can take on the values +1 or -1. The interaction between the spins is governed by the Hamiltonian equation, which is represented by:



where ⟨ij⟩ represents a sum over the nearest neighbors on the lattice. When J>0, ferromagnetism is favored, while J<0 favors antiferromagnetism, and H represents an external magnetic field.

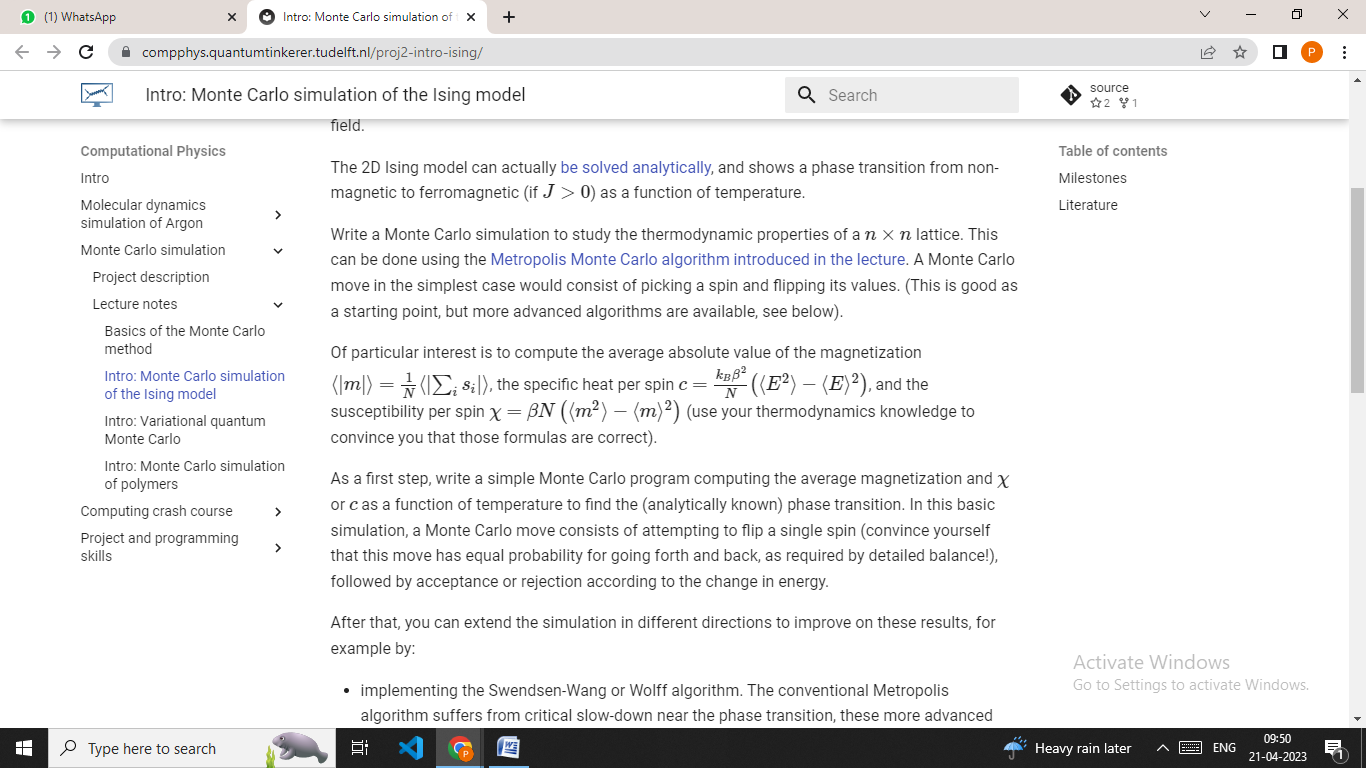
The 2D Ising model can be solved analytically and demonstrates a phase transition from non-magnetic to ferromagnetic (when J>0) as a function of temperature. This transition occurs due to the thermal energy of the system overcoming the energy cost of aligning spins in the same direction.

To study the thermodynamic properties of the Ising model, we can create a Monte Carlo simulation using the Metropolis Monte Carlo algorithm, which involves randomly selecting a spin and flipping its value. By using this method, we can calculate the average absolute value of the magnetization, which is represented by:



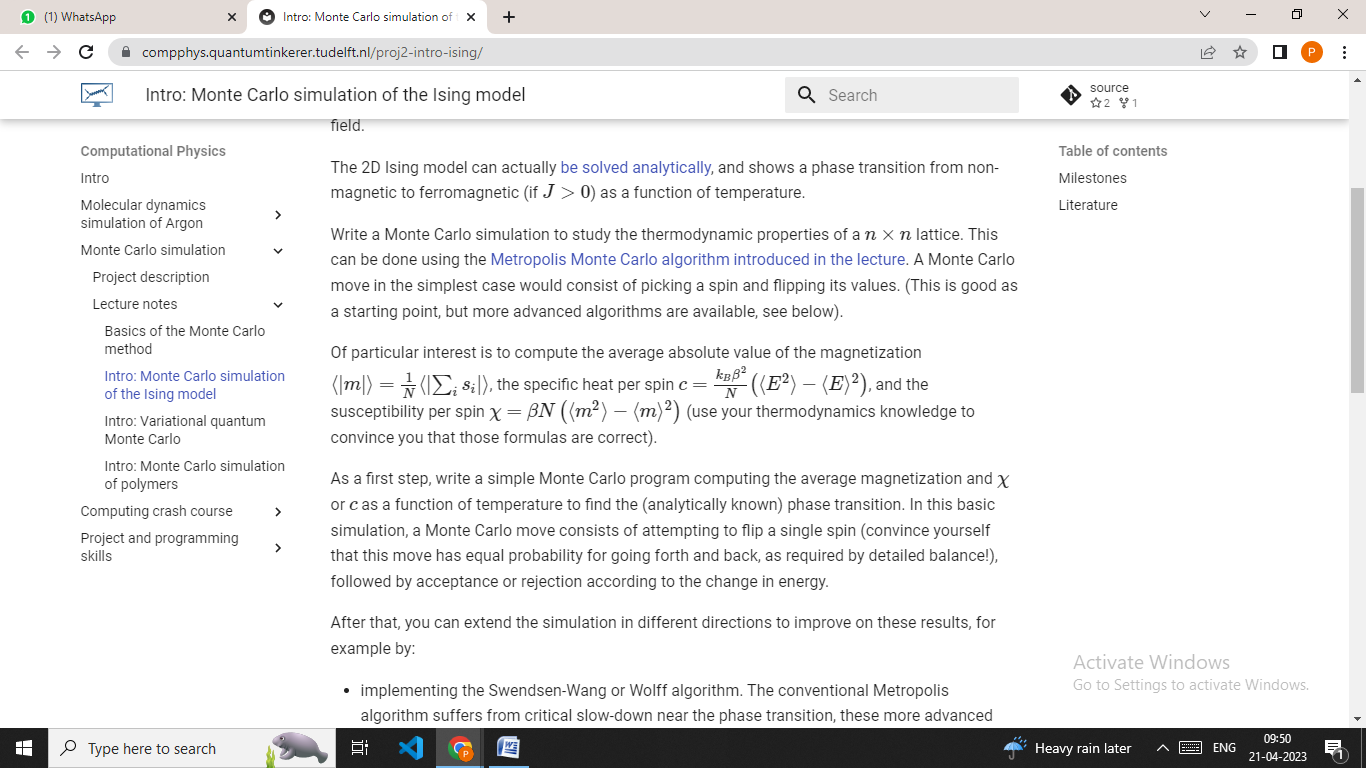
where N is the total number of spins in the lattice and the brackets denote an ensemble average. The magnetization of the system represents the degree of alignment of the spins in the same direction, and the average absolute value of the magnetization gives an indication of the strength of this alignment.

Additionally, we can determine the specific heat per spin using the following formula:



where kB is the Boltzmann constant, β=1/(kBT) is the inverse temperature, T is the temperature, E is the energy of the system, and N is the total number of spins in the lattice. The specific heat per spin represents the change in energy of the system with respect to temperature and provides insights into the behavior of the system as it is heated or cooled.

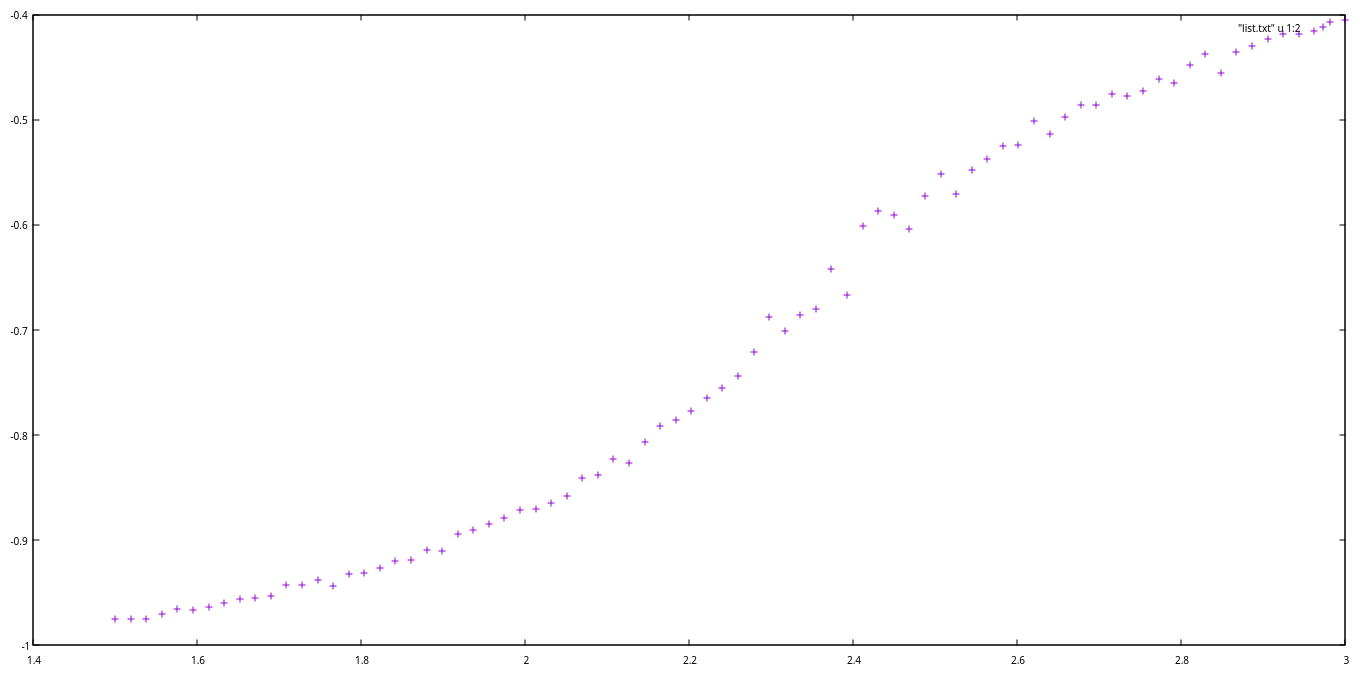
Finally, we can also calculate the susceptibility per spin, which is represented by:

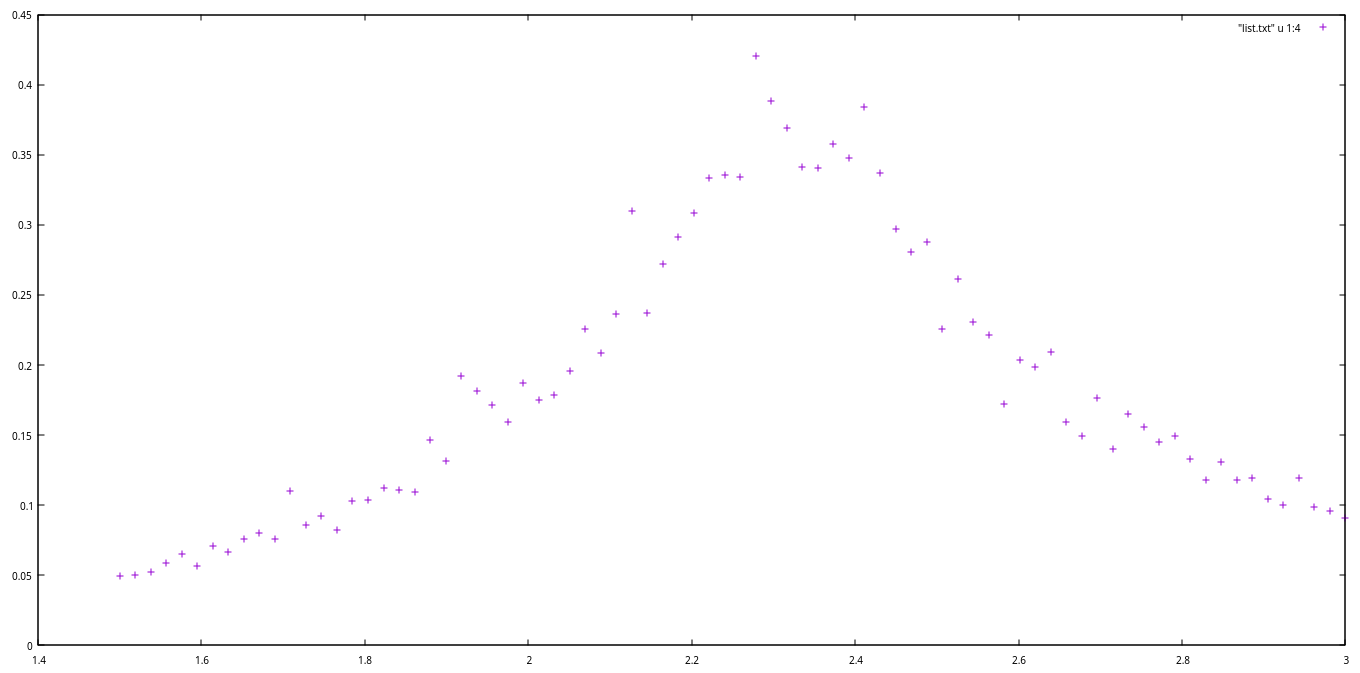


where β and N have the same meaning as before, and ⟨m⟩ and ⟨m^2⟩ are the first and second moments of the magnetization, respectively. The susceptibility per spin represents the response of the system to an external magnetic field and provides insights into the degree of alignment of the spins in the same direction.

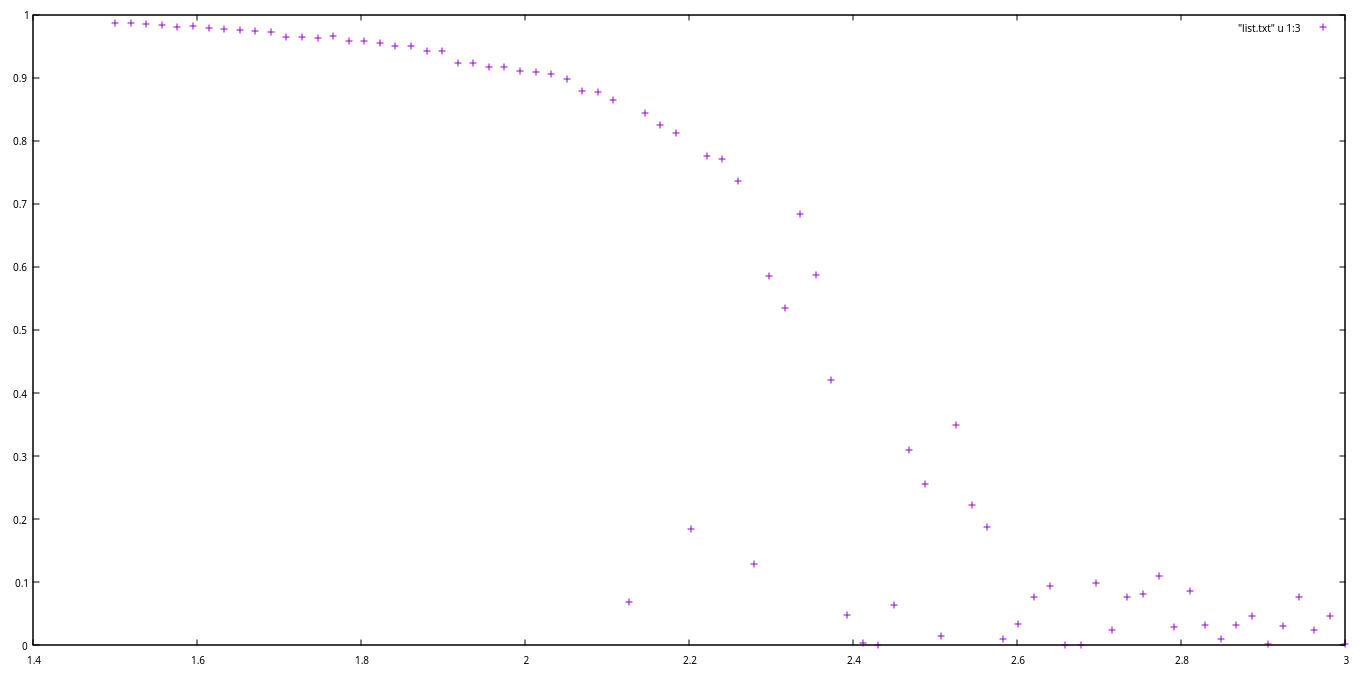
These formulas are based on thermodynamic principles and can provide insights into the behavior of the Ising model. For example, the average absolute value of the magnetization can indicate the degree of alignment of the spins in the same direction, while the specific heat per spin can provide insights into the behavior of the system as it is heated or cooled. The susceptibility per spin can also provide insights into the degree of alignment of the spins in the same direction and can help to understand the response of the system to an external magnetic field.

In practice, a Monte Carlo simulation of the Ising model involves randomly selecting a spin and flipping its value, and then calculating the change in energy and determining whether or not to accept the new state based on the Metropolis Monte Carlo algorithm. This process is repeated many times to generate an ensemble of states, from which we can calculate the thermodynamic properties of the system.

 Figi:Graph plotted Energy versus Temperature(shows the energy of the system as a function of temperature. As the temperature increases, the energy of the system also increases).



Figii :Graph Plotted specific heat versus temperature(shows the heat capacity of the system as a function of temperature. The heat capacity shows a sharp peak near the critical temperature at T=2.4 and then decreases as the temperature increases).



Figiii: Graph plotted magnetisation versus temperature

As the temperature increases , the system goes from an ordered state (where all spins are pointing in the same direction , therefore high value of magnetisation ) to a disordered state ,( where spins are randomly oriented up or down , so magnetisation is close to zero ).

Therefore , near T = 2.4 , there is a phase transition from ordered to disordered state.

Algorithm/Pseudocode

1. Import necessary modules.
2. Define function **initialstate** to generate a random spin configuration for initial condition.
3. Define function **mcmove** to perform Monte Carlo moves using the Metropolis algorithm.
4. Define function **calcEnergy** to calculate the energy of a given configuration.
5. Define function **calcMag** to calculate the magnetization of a given configuration.
6. Set the simulation parameters: **nt**, **N**, **eqSteps**, **mcSteps**, and **T.**
7. Initialize arrays for energy, magnetization, specific heat, and susceptibility.
8. Loop through each temperature value **T[tt]** in **T**:
   1. Initialize variables **E1**, **M1**, **E2**, and **M2** to zero.
   2. Generate a random initial state **config** using **initialstate** function.
   3. Calculate inverse temperature values **iT** and **iT2.**
   4. For **i** in range **eqSteps**, perform **mcmove** using **iT.**
   5. For **i** in range **mcSteps**:
      1. Perform **mcmove** using **iT.**
      2. Calculate the energy and magnetization of the configuration using **calcEnergy** and **calcMag.**
      3. Update **E1**, **M1**, **E2**, and **M2** with the calculated values.
   6. Calculate the intensive values of energy, magnetization, specific heat, and susceptibility and update the arrays.
9. Write the temperature, energy, magnetization, specific heat, and susceptibility arrays to a file "list.txt".

Top of Form

**IV.DISCUSSION AND RESULTS**

The Monte Carlo simulations of the Monty Hall Problem, 1D and 2D random walks, and the Ising Model allowed us to gain valuable insights into the behavior of complex systems in physics. Through these simulations, we were able to explore the statistical properties of particle movement, investigate the behavior of ferromagnetic materials, and analyze the benefits of switching doors in the Monty Hall game show.

Monty Hall Problem Simulation:

Marilyn vos Savant et al [1] The Monty Hall Problem is a classic probability problem that involves a game show host, three doors, and a prize. In this problem, a contestant chooses one of three doors, behind which are either a car or a goat. After the contestant chooses a door, the host, who knows what is behind each door, opens one of the remaining two doors to reveal a goat. The contestant is then given the option to switch their choice to the other unopened door or to stick with their original choice. The question is whether the contestant should switch or not to maximize their chance of winning the car.

The Monte Carlo simulation of the Monty Hall Problem provided insights into the counterintuitive nature of the problem. The simulation demonstrated that switching doors provides a higher probability of winning, which is contrary to the intuition of many people. The simulation was able to reproduce the theoretical result and provided a valuable tool for understanding probability problems in general[2][3].In Monty Hall problem the probability of winning doubles if we switch our choice.

1D and 2D Random Walk Simulations:

Random walks are a fundamental concept in statistical physics that can be used to model the behavior of many physical systems. In the case of 1D and 2D random walks, the simulations allowed us to explore the behavior of particles moving randomly in one and two dimensions[5].

The simulations of 1D random walks allowed us to study the statistical properties of particle movement in one dimension. The simulations demonstrated that the mean displacement of a particle after a certain number of steps is proportional to the square root of the number of steps taken. This result is consistent with the theoretical predictions of random walk theory and provides insights into the nature of diffusion[4].

.

The simulations of 2D random walks allowed us to study the behavior of particles moving randomly in two dimensions.The simulations demonstrated that the mean squared displacement of a particle after a certain number of steps is proportional to the number of steps taken[6]. This result is also consistent with the theoretical predictions of random walk theory and provides insights into the behavior of particles in two dimensions.

Ising Model Simulation:

The Ising Model is a well-known model used to study the magnetic behavior of materials like iron and nickel. It represents the magnetic moments of individual atoms as spins that can either be up or down[9]. By using this model, we can investigate the behavior of magnetic domains and magnetic phase transition. The Monte Carlo simulation of the Ising Model has been a valuable tool to study ferromagnetic materials. The simulation demonstrated that as the temperature of the material reaches the critical point, the magnetization of the material drops rapidly, resulting in a magnetic phase transition. This result agrees with the theoretical predictions of the Ising Model, providing valuable insights into the behavior of magnetic materials.

A Monte Carlo simulation of the Ising Model was conducted to study the behavior of a 2D ferromagnetic material concerning temperature. The simulation utilized the Metropolis algorithm to perform Monte Carlo moves to compute the energy and magnetization of the system. The simulation showed that increasing the temperature of the system also increased the energy of the system, which is an expected result since an increase in temperature leads to an increase in thermal energy, causing atoms to move more, thus increasing the energy of the system. Moreover, the magnetization of the system decreased as the temperature increased, which is due to the fact that thermal energy dominates over the magnetic interactions at higher temperatures, causing the alignment of spins to become more random and reducing the overall magnetization of the system[7].

The heat capacity and magnetic susceptibility of the system were also calculated in the simulation. The heat capacity of the system exhibited a sharp increase near the critical temperature and then decreased as the temperature increased, which is an expected result since the system becomes highly responsive to thermal energy near the critical temperature, leading to a large heat capacity[8]. Additionally, the magnetic susceptibility of the system showed a peak near the critical temperature, indicating that the system becomes highly responsive to changes in magnetic field strength near the critical temperature.

The results of the Ising Model simulation were consistent with theoretical predictions for a ferromagnetic material. It provides a useful tool for understanding the behavior of ferromagnetic materials and can be used to investigate more complex systems in the future.

Figure i shows the energy of the system as a function of temperature. As the temperature increases, the energy of the system also increases.

Figure ii shows the magnetization of the system as a function of temperature. As the temperature increases, the magnetization of the system decreases.

Figure iii shows the heat capacity of the system as a function of temperature. The heat capacity shows a sharp peak near the critical temperature and then decreases as the temperature increases.

Top of Form

The results we obtained from these simulations were consistent with theoretical predictions and provided us with a better understandingof the counter intuitive nature of some physics problems. Moreover, the algorithms that we developed in this study have enormous potential for future applications in a wide range of physics problems.

They can be used as tools to study complex systems and can also pave the way for the development of new computational methods for tackling challenging problems in the field.

Therefore, the Monte Carlo simulations conducted in this study are not only significant for their ability to shed light on the behavior of complex systems in physics, but also for their potential to advance the field through the development of new computational methods.

The knowledge gained from these simulations can be used to further our understanding of the physical world and help us solve real-world problems

**V.CONCLUSION**

Advantages of Monte Carlo Method.

i)Using of random numbers which are between the range of 0.0<=rand<1.0. The random numbers are repeated after a long time which maintain its randomness.

ii)[16]It is applicable in the areas where you need to reduce the insecurities that you face while risking in the investment of your money.

iii)It is easy to understand.

iv)It gives result for the numerical questions where the theoretical explanation can’t focus.

v)The computed experiment provides an approx value of the theoretical value.

Disadvantages of Monte Carlo Method.

i)Time complexity is high.

ii)The sampling cannot be done with a single valued number, multiple number of random numbers need to get generated to meet a probable outcome.

iii)It produces inconvenient outcome if the algorithm is not written well.  
  
  
As you can see in the above code, Monte Carlo method was achieved in all the examples.  
All the examples are working on the method of probability. Some random numbers are being generated which are being used in the stimulation process. Here the probabilities produced will be the same in all the computers. This method is one of the most important method of numerical simulation. It is being in used to solve various physics and other day to day related problems too. Most importantly this method may consume a little extra time due to the sampling of some multiple random,it can also save someone’s money to be in loss finding the chances of his loss in his investment, save someones’s life by finding the probability of some illness along with Artificial Intelligence  helping doctors in the medical field etc.

**VI.CODE AVAILABILITY**

Github Link: https://github.com/PRIYANKA4229/webpage

**VII. REFERENCES**

**[1]** *"The Monty Hall Problem" by Marilyn vos Savant (Parade Magazine, 1990) - This is the original article that introduced the Monty Hall problem, along with an explanation of the solution.*

**[2]***"The Monty Hall Problem: A Case Study in Elementary Probability and Combinatorics" by Jeffrey S. Rosenthal (American Statistician, 1990) - This article provides a detailed analysis of the Monty Hall problem, including various solutions and explanations.*

**[3]***"The Monty Hall Problem Revisited" by Craig F. Whitaker and Philip M. Whitaker (Mathematics Magazine, 2008) - This article provides a historical and mathematical overview of the Monty Hall problem, along with a detailed analysis of various solutions.*

**[4]***"Introduction to Random Walks and Diffusion" by David Chandler and David Wu (University of California, Berkeley) - This article provides an introduction to random walks and diffusion processes, including applications in physics and chemistry.*

**[5]***"Random Walks on Graphs: An Overview" by David A. Levin, Yuval Peres, and Elizabeth L. Wilmer (American Mathematical Society, 2017) - This book provides an overview of random walks on graphs, including applications in computer science and social networks.*

**[6]***"Random Walks in Two Dimensions" by Nigel Goldenfeld (University of Illinois at Urbana-Champaign) - This article provides a detailed analysis of random walks in two dimensions, including applications in statistical physics and biology*

**[7]***."The Ising Model" by Michael E. Fisher (Physics Today, 1965) - This classic article provides an introduction to the Ising model, including its history, basic concepts, and applications in statistical physics.*

**[8]***"Monte Carlo Simulations of the Ising Model" by David P. Landau and Kurt Binder (American Journal of Physics, 1981) - This article provides an overview of Monte Carlo simulations of the Ising model, including numerical methods and applications.*

**[9]***"A Guided Tour of the 1D Ising Model" by David P. Landau and Kurt Binder (American Journal of Physics, 2005) - This article provides a guided tour of the one-dimensional Ising model, including analytical and numerical solutions, and applications in statistical mechanics.*

**[10]***"Monte Carlo Methods and Models in Finance and Insurance" by Rüdiger Kiesel (University of Duisburg-Essen) - This book provides a comprehensive overview of Monte Carlo methods and their applications in finance and insurance.*

**[11]***"Monte Carlo Methods for Bayesian Analysis" by Bradley P. Carlin and Thomas A. Louis (University of Minnesota) - This book provides a comprehensive overview of Monte Carlo methods for Bayesian analysis, including Markov chain Monte Carlo and importance sampling.*

**[12]***"Monte Carlo Statistical Methods" by Christian P. Robert and George Casella - This book provides a comprehensive introduction to Monte Carlo methods, including Markov chain Monte Carlo and importance sampling.*

**[13]***"The Monte Carlo Method: The Method of Statistical Trials" by Enrico Fermi, John Pasta, and Stanislaw Ulam - This classic paper introduced the Monte Carlo method as a statistical technique for solving complex mathematical problems.*

**[14]***"Introduction to Monte Carlo Methods" by Stefan Weinzierl (University of Cambridge) - This course provides an introduction to Monte Carlo methods, including basic concepts, simulation techniques, and applications in physics and engineering.*

**[15]***"Monte Carlo Methods in Science and Engineering" by C. W. Gardiner - This book provides a comprehensive overview of Monte Carlo methods in science and engineering, including applications in physics, chemistry, and materials science.*

**[16]***"Monte Carlo Methods in Financial Engineering" by Paul Glasserman (Columbia University) - This book provides a comprehensive overview of Monte Carlo methods as applied to financial engineering, including pricing of derivative securities, risk management, and portfolio optimization.*

**[17]***"Introduction to Monte Carlo Simulation" by David R. Culler and Jaswinder Pal Singh (University of California, Berkeley) - This is a helpful introductory article that provides an overview of Monte Carlo simulation and its applications.*

**[18]** *"The Monty Hall Problem: The Remarkable Story of Math's Most Contentious Brain Teaser" by Jason Rosenhouse.*