SATISH CHANDRA MEMORIAL SCHOOL

COMPUTER SCIENCE (083)

FINAL PROJECT (2020-2021)

PROJECT : TIC-TAC-TOE

SATISH CHANDRA MEMORIAL SCHOOL
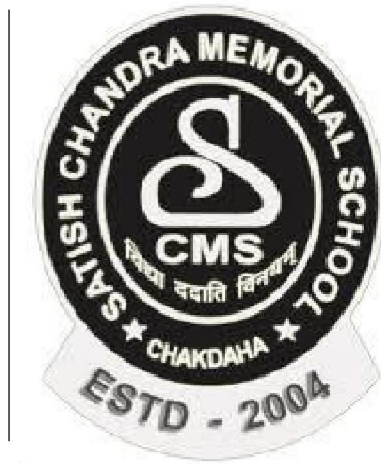
CERTIFICATE

This is to certify that Cadet _____

Roll No:_____has successfully completed the project

Work entitled **"TIC-TAC-TOE GAME."** in the subject Computer Science (083) laid

down in the regulations of CBSE for the purpose of Practical [Examination in

Class XI to be held in _____on_____.

( )
PGT Comp Sci
Master IC

Examiner:

Name:_____

Signature:

Date:

	TABLE OF CONTENTS [T O C]	
<u>**SER**</u>	<u>**DESCRIPTION**</u>	<u>**PAGE NO**</u>

ACKNOWLEDGEMENT

Apart from the efforts of me, the success of any project depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project.

I express my heartfelt gratitude to my parents for constant encouragement while carrying out this project.

I gratefully acknowledge the contribution of the individuals who contributed in bringing this project up to this level, who continues to look after me despite my flaws,

I express my deep sense of gratitude to the luminary **The Principal,** Arup Sarkar sir who has been continuously motivating and extending their helping hand to us.

I am overwhelmed to express my thanks to my father for providing me an infrastructure and moral support while carrying out this project in the school.

My sincere thanks to Debabrata Patikar sir , Master In-charge, A guide, Mentor all the above a friend, who critically reviewed my project and helped in solving each and every problem, occurred during implementation of the project

The guidance and support received from all the members (**Aryan Singh**, **Debopriyo Dalal**,**Rudrapalash**,**Rajarshi Saha**, and **Subhajit Kumar Das)** who contributed and who are contributing to this project, was vital for the success of the project. I am grateful for their constant support and help.

<h1 style="text-align:center">PROJECT ON <u>TIC-TAC-TOE GAME</u></h1>

<u>INTRODUCTION</u>

Tic-tac-toe (American English), **noughts and crosses** (British English), or **Xs and Os** is a paper-and-pencil game for two players, *X* and *O*, who take turns marking the spaces in a 3×3 grid. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row wins the game.

<u>OBJECTIVES OF THE PROJECT</u>

The objective of this project is to let the students apply the programming knowledge into a real- world situation and solve problems that one faces during programming.

1.   Expose the students to implement their theoretical knowledge of programming to build programs for practical application.

2.   Enable students to better understand programming concept in an interactive way and enable them to write efficient codes for their programs.

3.   Enable students to solve problems by coding.

4.   Improve the skills to present work of a student using computer.

5.   Expose the students to the real world problems that they may face while writing a program source code so that they may learn to handle such situations on their own.

PROPOSED SYSTEM

Today one cannot afford to rely on the fallible human beings of be really wants to stand against today's merciless competition where not to wise saying **"to err is human"** no longer valid, it's outdated to rationalize your mistake. So, to keep pace with time, to bring about the best result without malfunctioning and greater efficiency so to replace the unending heaps of flies with a much sophisticated hard disk of the computer.

One has to use the data management software. Software has been an ascent in atomization various organisations. Many software products working are now in markets, which have helped in making the organizations work easier and efficiently. Data management initially had to maintain a lot of ledgers and a lot of paper work has to be done but now software product on this organization has made their work faster and easier. Now only this software has to be loaded on the computer and work can be done.

This prevents a lot of time and money. The work becomes fully automated and any information regarding the organization can be obtained by clicking the button. Moreover, now it's an age of computers of and automating such an organization gives the better look.

SYSTEM DEVELOPMENT LIFE CYCLE (SDLC)
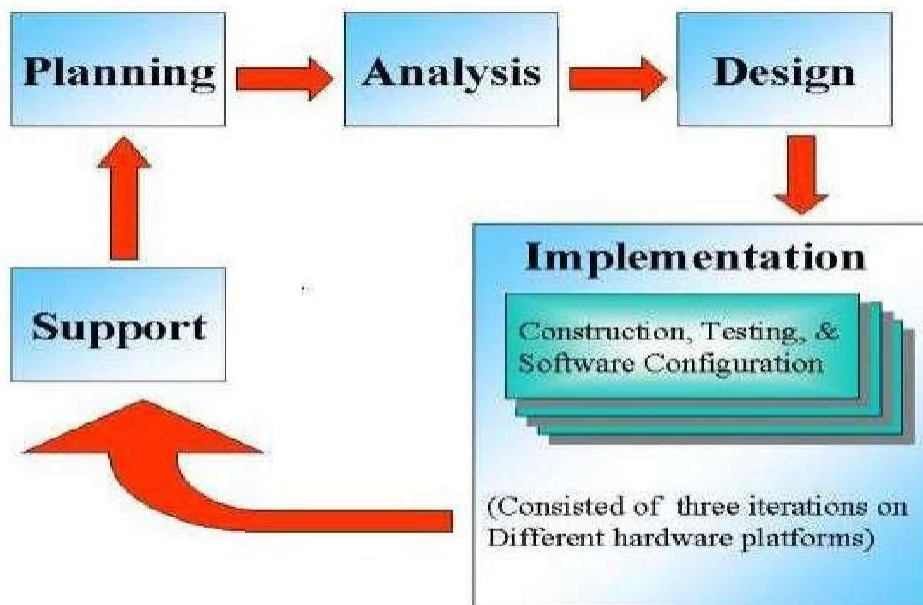
The systems development life cycle is a project management technique that divides complex projects into smaller, more easily managed segments or phases. Segmenting projects allows managers to verify the successful completion of project phases before allocating resources to subsequent phases.

Software development projects typically include initiation, planning, design, development, testing, implementation, and maintenance phases. However, the phases may be divided differently depending on the organization involved.

For example, initial project activities might be designated as request, requirements-definition, and planning phases, or initiation, concept-development, and planning phases. End users of the system under development should be involved in reviewing the output of each phase to ensure the system is being built to deliver the needed functionality.

PHASES OF SYSTEM DEVELOPMENT LIFE CYCLE

INITIATION PHASE

The Initiation Phase begins when a business sponsor identifies a need or an opportunity.

The purpose of the Initiation Phase is to:

- Identify and validate an opportunity to improve business accomplishments of the organization or a deficiency related to a business need.
- Identify significant assumptions and constraints on solutions to that need.
- Recommend the exploration of alternative concepts and methods to satisfy the need including questioning the need for technology, i.e., will a change in the business process offer a solution?
- Assure executive business and executive technical sponsorship. The Sponsor designates a Project Manager and the business need is documented in a Concept Proposal. The Concept Proposal includes information about the business process andthe relationship to the Agency/Organization.
- Infrastructure and the Strategic Plan. A successful Concept Proposal results in a Project Management Charter which outlines the authority of the project manager to begin the project.

Careful oversight is required to ensure projects support strategic business objectives and resources are effectively implemented into an organization's enterprise architecture. The initiation phase begins when an opportunity to add, improve, or correct a system is identified and formally requested through the presentation of a business case. The business case should, at a minimum, describe a proposal's purpose, identify expected benefits, and explain how the proposed system supports one of the organization's business strategies. The business case should also identify alternative solutions and detail as many informational, functional, and network requirements as possible.

SYSTEM CONCEPT DEVELOPMENT PHASE

The System Concept Development Phase begins after a business need or opportunity is validated by the Agency/Organization Program Leadership and the Agency/Organization CIO.

The purpose of the System Concept Development Phase is to:

- Determine the feasibility and appropriateness of the alternatives.
- Identify system interfaces.
- Identify basic functional and data requirements to satisfy the business need.
- Establish system boundaries; identify goals, objectives, critical success factors, and performance measures.
- Evaluate costs and benefits of alternative approaches to satisfy the basic functional requirements
- Assess project risks
- Identify and initiate risk mitigation actions, andDevelop high-level technical architecture, process models, data models, and a concept of operations. This phase explores potential technical solutions within the context of the business need.
- It may include several trade-off decisions such as the decision to use COTS software products as opposed to developing custom software or reusing software components, or the decision to use an incremental delivery versus a complete, onetime deployment.
- Construction of executable prototypes is encouraged to evaluate technology to support the business process. The System Boundary Document serves as an important reference document to support the Information Technology Project Request (ITPR) process.
- The ITPR must be approved by the State CIO before the project can move forward.

PICTORIAL REPRESENTATION OF SDLC:
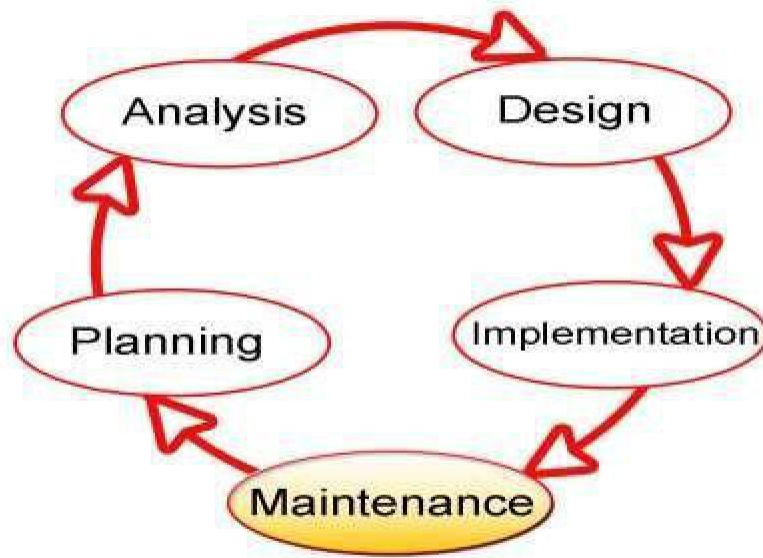
PLANNING PHASE

The planning phase is the most critical step in completing development, acquisition, and maintenance projects. Careful planning, particularly in the early stages of a project, is necessary to coordinate activities and manage project risks effectively. The depth and formality of project plans should be commensurate with the characteristics and risks of a given project. Project plans refine the information gathered during the initiation phase by further identifying the specific activities and resources required to complete a project.

A critical part of a project manager' sjob is to coordinate discussions between user, audit, security, design, development, and network personnel to identify and document as many functional, security, and network requirements as possible. During this phase, a plan is developed that documents the approach to be used and includes a discussion of methods, tools, tasks, resources, project schedules, and user input. Personnel assignments, costs, project schedule, and target dates are established.

A Project Management Plan is created with components related to acquisition planning, configuration management planning, quality assurance planning, concept

of operations, system security, verification and validation, and systems engineering management planning.

REQUIREMENTS ANALYSIS PHASE

This phase formally defines the detailed functional user requirements using high-level requirements identified in the Initiation, System Concept, and Planning phases. It also delineates the requirements in terms of data, system performance, security, and maintainability requirements for the system. The requirements are defined in this phase to alevel of detail sufficient for systems design to proceed. They need to be measurable, testable, and relate to the business need or opportunity identified in the Initiation Phase. The requirements that will be used to determine acceptance of the system are captured in the Test and Evaluation MasterPlan.

The purposes of this phase are to:

- Further define and refine the functional and data requirements and document them in the Requirements Document,
- Complete business process reengineering of the functions to be supported (i.e., verify what information drives the business process, what information is generated, who generates it, where does the information go, and who processes it),
- Develop detailed data and process models (system inputs, outputs, and the process.
- Develop the test and evaluation requirements that will be used to determine acceptable system performance**.**

DESIGN PHASE

The design phase involves converting the informational, functional, and network requirements identified during the initiation and planning phases into unified design specifications that developers use to scriptprograms during the development phase. Program designs are c onstructed in various ways. Using a top-down approach, designers first identify and link majorprogram components and interfaces, then expand design layouts as they identify and link smaller subsystems and

connections. Using a bottom-up approach, designers first identify and link minor program components and interfaces, then expand design layouts as they identify and link larger systems and connections. Contemporary design techniques often use prototyping tools that build mock-up designs of items such as application screens, database layouts, and system architectures. End users, designers, developers, database managers, and network administrators should review and refine the prototyped designs in an iterative process until they agree on an acceptable design. Audit, security, and quality assurance personnel should be involved in the review and approval process. During this phase, the system is designed to satisfy the functional requirements identified in the previous phase. Since problems in the design phase could be very expensive to solve in the later stage of the software development, a variety of elements are considered in the design to mitigate risk. These include:

- Identifying potential risks and defining mitigating design features.
- Performing a security risk assessment.
- Developing a conversion plan to migrate current data to the new system.
- Determining the operating environment.
- Defining major subsystems and their inputs and outputs.
- Allocating processes to resources.
- Preparing detailed logic specifications for each software module. The result is a draft System Design Document which captures the preliminary design for the system.
- Everything requiring user input or approval is documented and reviewed by the user. Once these documents have been approved by the Agency CIO and Business Sponsor, the final System Design Document is created to serve as the Critical/Detailed Design for the system.
- This document receives a rigorous review byAgency technical and functional representatives to ensure that it satisfies the business requirements. Concurrent with the development of the system design, the Agency Project Manager begins development of the Implementation Plan, Operations and Maintenance Manual, and the Training Plan.

DEVELOPMENT PHASE

The development phase involves converting design specifications into executable programs. Effective development standards include requirements that programmers and other project participants discuss design specifications before programming begins. The procedures help ensure programmers clearly understand program designs and functional requirements. Programmers use various techniques to develop computer programs. The large transaction oriented programs associated with financial institutions have traditionally been developed using procedural programming techniques. Procedural programming involves the line-by-line scripting of logical instructions that are combined to form a program. Effective completion of the previous stages is a key factor in the success of the Development phase. The Development phase consists of:

- Translating the detailed requirements and design into system components.
- Testing individual elements (units) for usability.
- Preparing for integration and testing of the IT system.

INTEGRATION AND TEST PHASE

- Subsystem integration, system, security, and user acceptance testing is conducted during the integration and test phase. The user, with those responsible for quality assurance, validates that the functional requirements, as defined in the functional requirements document, are satisfied by the developed or modified system. OIT Security staff assess the system security and issue a security certification and accreditation prior to installation/implementation.

Multiple levels of testing are performed, including:

- Testing at the development facility by the contractor and possibly supported by end users

- Testing as a deployed system with end users working together with contract personnel

- Operational testing by the end user alone performing all functions. Requirements are traced throughout testing,a final Independent Verification & Validation evaluation is performed and all documentation is reviewedand accepted prior to acceptance of the system.

IMPLEMENTATION PHASE

This phase is initiated after the system has been tested and accepted by the user. In this phase, the system is installed to support the intended business functions. System performance is compared to performance objectives established during the planning phase. Implementation includes user notification, user training, installation of hardware, installation of software onto production computers, and integration of the system into daily work processes. This phase continues until the system is operating in production in accordance with the defined userrequirements.

OPERATIONS AND MAINTENANCE PHASE

The system operation is ongoing. The system is monitored for continued performance in accordance with user requirements and needed system modifications are incorporated. Operations continue as long as the system can be effectively adapted to respond to the organization's needs. When modifications or changes are identified, the system may reenter the planning phase.

The purpose of this phase is to:

- Operate, maintain, and enhance the system.
- Certify that the system can process sensitive information.
- Conduct periodic assessments of the system to ensure the functional requirements continue to be satisfied.
- Determine when the system needs to be modernized, replaced, or retired.

FLOW CHART
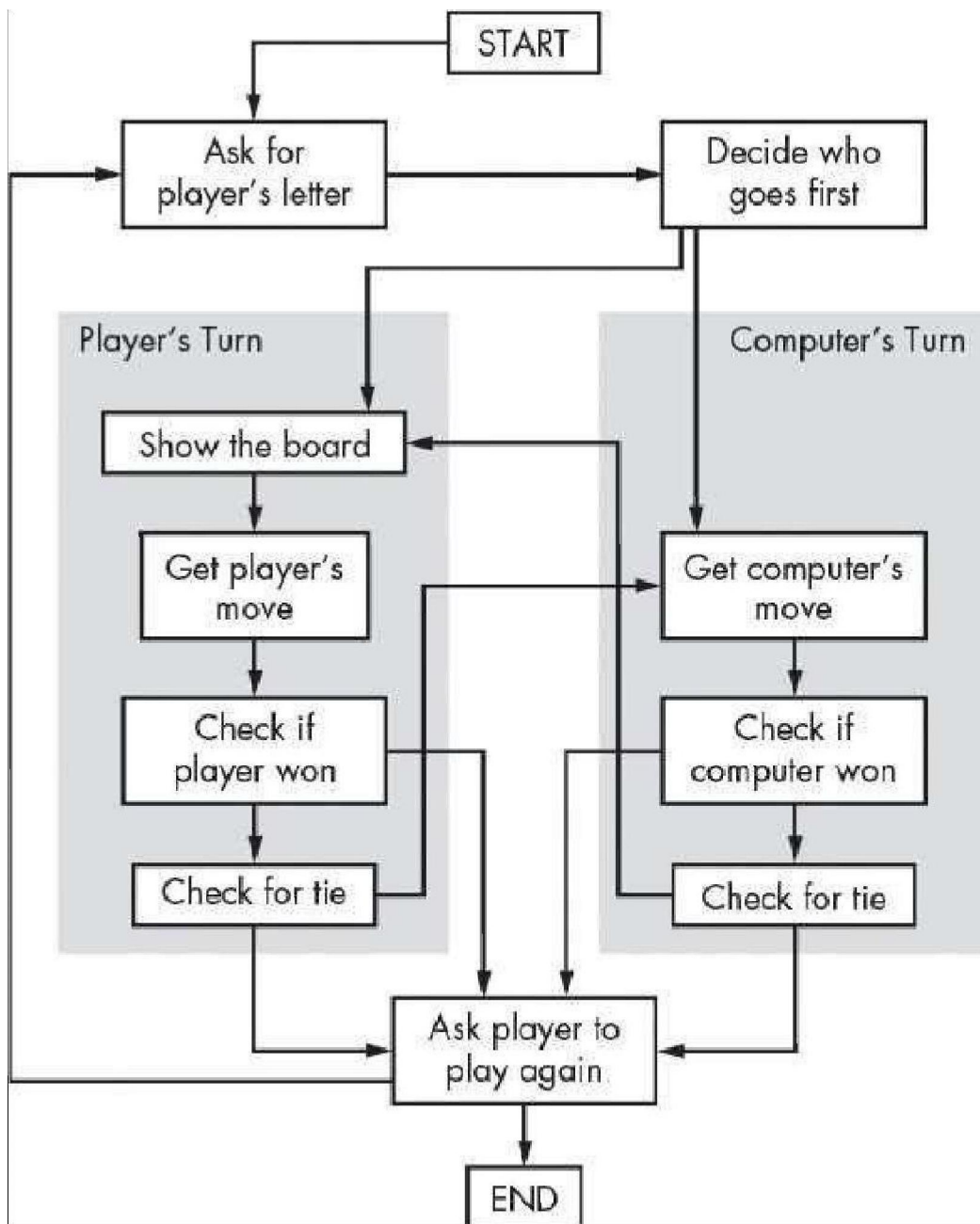
```python
import
random
        data_dic = {'a': " ", 'b': " ", 'c': " ", 'd': " ", 'e': " ", 'f': " ", 'g': " ", 'h': " ", 'i': " "}
        Choice_dict = {'Player 1' : " ", 'Player 2' : " ", 'Computer': " "}
        var = random.choice(['X','O','$','#','%','&','@'])
        GREEN = '\u001b[32m'
        YELLOW = '\u001b[33m'
        RESET = '\u001b[0m'
        RED = '\u001b[31m'
        CYAN = '\u001b[36m'
        MAGENTA = '\u001b[35m'
        BLUE = '\u001b[34m'
        BOLD = '\u001b[1m'



        def Notice():
            print("""Welcome to the Tic Tac Toe Game,
                                    by Apurba Ghosh
This Game involves two Functionality, 1. Player vs Player
                                      2. Player vs Computer
Choose an Option from 1 or 2 to Start the game, There are some
functionalities to be noticed, if you want to Quit the game at
any point of time, just type ('exit',) as input,
Rules:
    1) You can't choose a column already filled, doing that it will
    give a prompt as many times as you do that.

    2) Type your name at the very first starting of the program.

    3) Avoid Choosing the same variable for both the user, if played
    Player vs Player.

    4) You will always have to Give the same Variable which you have
    given as the very first input at the starting of the game, note:
```

```python
        that you are free to choose any Variables or even numbers and not
        limited to only (X/O). Henceforth, Freedom is granted.
              \n\n""")


def choice():
    x = int(input("""1. Player vs Player
2. Player vs Computer \t Choose: """))
    return (x)


def board(container : str = "", screenwidth : int = 59, sign1 = " ", sign2 = " ", sign3 = " "):
    counter = 0
    while (counter <= 21):
        if (counter == 0 or counter == 14 or counter == 7 or counter == 21):
            container = f"{YELLOW}-{RESET}" * screenwidth
            print(container)
            counter += 1
            continue

        if (counter == 3):
            container = f"{YELLOW}|{RESET}          {data_dic['a']}        {YELLOW}|{RESET}          {data_dic['b']}        {YELLOW}|{RESET}
{data_dic['c']}        {YELLOW}|{RESET}" f"{sign1}"
            print(container)
            counter += 1
            continue

        if (counter == 10):
            container = f"{YELLOW}|{RESET}          {data_dic['d']}        {YELLOW}|{RESET}          {data_dic['e']}        {YELLOW}|{RESET}
{data_dic['f']}        {YELLOW}|{RESET}" f"{sign2}"
            print(container)
            counter += 1
            continue
```

```python
        if (counter == 17):
            container = f"{YELLOW}|{RESET}          {data_dic['g']}       {YELLOW}|{RESET}          {data_dic['h']}       {YELLOW}|{RESET}      {data_dic['i']}      {YELLOW}|{RESET}" f"{sign3}"
            print(container)
            counter += 1
            continue


        else:
            container = f"{YELLOW}|                 |                 |                 |"
            print(container)
            counter += 1



def check(a : str ,b : str ,c : str ,d : str ,e : str ,f : str ,g : str ,h : str ,i : str ):

    # Part 1:
    if (a == b == c):
        return (a)
    if (d == e == f):
        return (d)
    if (g == h == i):
        return (g)

    # Part 2:
    if (a == d == g):
        return (a)
    if (b == e == h):
        return (b)
    if (c == f == i):
        return (c)

    # Part 3:
    if (a == e == i):
        return (a)
```

```python
        if (c == e == g):
            return (c)
        else:
            return (" ")



def computerOverwrite(user : str):
    if (data_dic[user] != " "):
        return (True)
    else:
        return (False)

def overwrite(user : tuple):
    if (data_dic[user[1]] != " "):
        print(f"{RED}Sorry the Column is already filled, Please Choose any other Column...{RESET}")
        return (True)
    else:
        return (False)



def game():
    Notice()
    choices = choice()
    name2 = ""
    if choices == 2:
        name1 = input('Player 1, Enter Your Name: ')
    else:
        name1 = input('Player 1, Enter Your Name: ')
        name2 = input('Player 2, Enter Your Name: ')
    board()
    winner = ""
    counter = 1
    continuous_check = " "
    f = ""
```

```python
while (counter <= 9):
    if (counter%2 != 0):
        user1 = eval(input("Player 1, Enter the Cloumn: "))
        if (user1[0] == 'exit'):
            print(f"{BLUE}Thanks For Playing{RESET}")
            f = 'a'
            break
        if (Choice_dict['Player 1'] == " "):
            pass
        else:
            if (Choice_dict['Player 1'] == user1[0]):
                pass
            else:
                print(f"{RED}Sorry You can't take {user1[0]}, You have to choose {Choice_dict['Player 1']}{RESET}")
                user1 = eval(input("Player 1, Enter the Cloumn: "))
                if (user1[0] == 'exit'):
                    print(f"{BLUE}Thanks For Playing{RESET}")
                    f = 'a'
                    break
                while (user1[0] != Choice_dict['Player 1']):
                    print(f"{RED}Sorry You can't take {user1[0]}, You have to choose {Choice_dict['Player 1']}{RESET}")
                    user1 = eval(input("Player 1, Enter the Cloumn: "))
                    if (user1[0] == 'exit'):
                        print(f"{BLUE}Thanks For Playing{RESET}")
                        f = 'a'
                        break
                if (user1[0] == 'exit'):
                    f = 'a'
                    break
                else:
                    pass
        info = overwrite(user1)
        if (info == True):
            user1 = eval(input("Player 1, Enter the Cloumn: "))
```

```python
            if (user1[0] == 'exit'):
                print(f"{BLUE}Thanks For Playing{RESET}")
                f = 'a'
                break
        while (user1[0] != Choice_dict['Player 1']):
            print(f"{RED}Sorry You can't take {user1[0]}, You have to choose {Choice_dict['Player 1']}{RESET}")
            user1 = eval(input("Player 1, Enter the Cloumn: "))
            if (user1[0] == 'exit'):
                print(f"{BLUE}Thanks For Playing{RESET}")
                f = 'a'
                break
        if (user1[0] == 'exit'):
            f = 'a'
            break
        info2 = overwrite(user1)
        while (info2 != False):
            user1 = eval(input("Player 1, Enter the Cloumn: "))
            if (user1[0] == 'exit'):
                print(f"{BLUE}Thanks For Playing{RESET}")
                f = 'a'
                break
            info2 = overwrite(user1)
            if (Choice_dict['Player 1'] == user1[0]):
                pass
            else:
                print(f"{RED}Sorry You can't take {user1[0]}, You have to choose {Choice_dict['Player 1']}{RESET}")
                user1 = eval(input("Player 1, Enter the Cloumn: "))
                if (user1[0] == 'exit'):
                    print(f"{BLUE}Thanks For Playing{RESET}")
                    f = 'a'
                    break
                while (user1[0] != Choice_dict['Player 1']):
                    print(f"{RED}Sorry You can't take {user1[0]}, You have to choose {Choice_dict['Player 1']}{RESET}")
                    user1 = eval(input("Player 1, Enter the Cloumn: "))
```

```python
            if (user1[0] == 'exit'):
                print(f"{BLUE}Thanks For Playing{RESET}")
                f = 'a'
                break
        info2 = overwrite(user1)

    if (user1[0] == 'exit'):
        f = 'a'
        break
    else:
        pass
else:
    pass


if (user1[1] == 'a'):
    data_dic['a'] = f"{RED}{user1[0]}{RESET}"
    board(sign1=f"{GREEN} <<--- {RESET}")
if (user1[1] == 'b'):
    data_dic['b'] = f"{RED}{user1[0]}{RESET}"
    board(sign1=f"{GREEN} <<--- {RESET}")
if (user1[1] == 'c'):
    data_dic['c'] = f"{RED}{user1[0]}{RESET}"
    board(sign1=f"{GREEN} <<--- {RESET}")
if (user1[1] == 'd'):
    data_dic['d'] = f"{RED}{user1[0]}{RESET}"
    board(sign2=f"{GREEN} <<--- {RESET}")
if (user1[1] == 'e'):
    data_dic['e'] = f"{RED}{user1[0]}{RESET}"
    board(sign2=f"{GREEN} <<--- {RESET}")
if (user1[1] == 'f'):
    data_dic['f'] = f"{RED}{user1[0]}{RESET}"
    board(sign2=f"{GREEN} <<--- {RESET}")
if (user1[1] == 'g'):
    data_dic['g'] = f"{RED}{user1[0]}{RESET}"
```

```python
                    board(sign3=f"{GREEN} <<--- {RESET}")
                if (user1[1] == 'h'):
                    data_dic['h'] = f"{RED}{user1[0]}{RESET}"
                    board(sign3=f"{GREEN} <<--- {RESET}")
                if (user1[1] == 'i'):
                    data_dic['i'] = f"{RED}{user1[0]}{RESET}"
                    board(sign3=f"{GREEN} <<--- {RESET}")
                Choice_dict['Player 1'] = user1[0]
                winner = check(data_dic['a'],data_dic['b'],data_dic['c'],data_dic['d'],data_dic['e'],data_dic['f']
                        ,data_dic['g'],data_dic['h'],data_dic['i'])
                if (winner != " "):
                    continuous_check = (True,winner)
                    break
        else:
            if choices == 2:
                print("Computer's Turn...")
                column = ''
                for j in range(1):
                    column = random.choice(['a','b','c','d','e','f','g','h','i'])
                    info = computerOverwrite(column)
                    while (info != False):
                        column = random.choice(['a','b','c','d','e','f','g','h','i'])
                        info = computerOverwrite(column)
                if (column == 'a'):
                    data_dic['a'] = f"{CYAN}{var}{RESET}"
                    board(sign1=f"{YELLOW} <<--- {RESET}")
                if (column == 'b'):
                    data_dic['b'] = f"{CYAN}{var}{RESET}"
                    board(sign1=f"{YELLOW} <<--- {RESET}")
                if (column == 'c'):
                    data_dic['c'] = f"{CYAN}{var}{RESET}"
                    board(sign1=f"{YELLOW} <<--- {RESET}")
                if (column == 'd'):
                    data_dic['d'] = f"{CYAN}{var}{RESET}"
```

```python
            board(sign2=f"{YELLOW} <<--- {RESET}")
        if (column == 'e'):
            data_dic['e'] = f"{CYAN}{var}{RESET}"
            board(sign2=f"{YELLOW} <<--- {RESET}")
        if (column == 'f'):
            data_dic['f'] = f"{CYAN}{var}{RESET}"
            board(sign2=f"{YELLOW} <<--- {RESET}")
        if (column == 'g'):
            data_dic['g'] = f"{CYAN}{var}{RESET}"
            board(sign3=f"{YELLOW} <<--- {RESET}")
        if (column == 'h'):
            data_dic['h'] = f"{CYAN}{var}{RESET}"
            board(sign3=f"{YELLOW} <<--- {RESET}")
        if (column == 'i'):
            data_dic['i'] = f"{CYAN}{var}{RESET}"
            board(sign3=f"{YELLOW} <<--- {RESET}")
        winner = check(data_dic['a'],data_dic['b'],data_dic['c'],data_dic['d'],data_dic['e'],data_dic['f']
            ,data_dic['g'],data_dic['h'],data_dic['i'])
        Choice_dict['Computer'] = var
        if (winner != " "):
            continuous_check = (True,winner)
            break
    else:
        user2 = eval(input("Player 2, Enter the Cloumn: "))
        if (user2[0] == 'exit'):
            print(f"{BLUE}Thanks For Playing{RESET}")
            f = 'b'
            break
        if (Choice_dict['Player 2'] == " "):
            pass
        else:
            if (Choice_dict['Player 2'] == user2[0]):
                pass
            else:
```

```python
                print(f"{RED}Sorry You can't take {user2[0]}, You have to choose {Choice_dict['Player 2']}{RESET}")
                user2 = eval(input("Player 2, Enter the Cloumn: "))
                if (user2[0] == 'exit'):
                    print(f"{BLUE}Thanks For Playing{RESET}")
                    f = 'b'
                    break
                while (user2[0] != Choice_dict['Player 2']):
                    print(f"{RED}Sorry You can't take {user2[0]}, You have to choose {Choice_dict['Player 2']}{RESET}")
                    user2 = eval(input("Player 2, Enter the Cloumn: "))
                    if (user2[0] == 'exit'):
                        print(f"{BLUE}Thanks For Playing{RESET}")
                        f = 'b'
                        break
                if (user2[0] == 'exit'):
                    f = 'b'
                    break
                else:
                    pass
    info = overwrite(user2)
    if (info == True):
        user2 = eval(input("Player 2, Enter the Cloumn: "))
        if (user2[0] == 'exit'):
            print(f"{BLUE}Thanks For Playing{RESET}")
            f = 'b'
            break
        while (user2[0] != Choice_dict['Player 2']):
            print(f"{RED}Sorry You can't take {user2[0]}, You have to choose {Choice_dict['Player 2']}{RESET}")
            user2 = eval(input("Player 2, Enter the Cloumn: "))
            if (user2[0] == 'exit'):
                print(f"{BLUE}Thanks For Playing{RESET}")
                f = 'b'
                break
        if (user2[0] == 'exit'):
            f = 'b'
```

```python
            break
        info2 = overwrite(user2)
        while (info2 != False):
            user2 = eval(input("Player 2, Enter the Cloumn: "))
            if (user2[0] == 'exit'):
                print(f"{BLUE}Thanks For Playing{RESET}")
                f = 'b'
                break
            info2 = overwrite(user2)
            if (Choice_dict['Player 2'] == user2[0]):
                pass
            else:
                print(f"{RED}Sorry You can't take {user2[0]}, You have to choose {Choice_dict['Player 2']}{RESET}")
                user2 = eval(input("Player 2, Enter the Cloumn: "))
                if (user2[0] == 'exit'):
                    print(f"{BLUE}Thanks For Playing{RESET}")
                    f = 'b'
                    break
                while (user2[0] != Choice_dict['Player 2']):
                    print(f"{RED}Sorry You can't take {user2[0]}, You have to choose {Choice_dict['Player 2']}{RESET}")
                    user2 = eval(input("Player 2, Enter the Cloumn: "))
                    if (user2[0] == 'exit'):
                        print(f"{BLUE}Thanks For Playing{RESET}")
                        f = 'b'
                        break
                info2 = overwrite(user2)

        if (user2[0] == 'exit'):
            f = 'b'
            break
        else:
            pass
    else:
        pass
```

```python
        if (user2[1] == 'a'):
            data_dic['a'] = f"{CYAN}{user2[0]}{RESET}"
            board(sign1=f"{YELLOW} <<--- {RESET}")
        if (user2[1] == 'b'):
            data_dic['b'] = f"{CYAN}{user2[0]}{RESET}"
            board(sign1=f"{YELLOW} <<--- {RESET}")
        if (user2[1] == 'c'):
            data_dic['c'] = f"{CYAN}{user2[0]}{RESET}"
            board(sign1=f"{YELLOW} <<--- {RESET}")
        if (user2[1] == 'd'):
            data_dic['d'] = f"{CYAN}{user2[0]}{RESET}"
            board(sign2=f"{YELLOW} <<--- {RESET}")
        if (user2[1] == 'e'):
            data_dic['e'] = f"{CYAN}{user2[0]}{RESET}"
            board(sign2=f"{YELLOW} <<--- {RESET}")
        if (user2[1] == 'f'):
            data_dic['f'] = f"{CYAN}{user2[0]}{RESET}"
            board(sign2=f"{YELLOW} <<--- {RESET}")
        if (user2[1] == 'g'):
            data_dic['g'] = f"{CYAN}{user2[0]}{RESET}"
            board(sign3=f"{YELLOW} <<--- {RESET}")
        if (user2[1] == 'h'):
            data_dic['h'] = f"{CYAN}{user2[0]}{RESET}"
            board(sign3=f"{YELLOW} <<--- {RESET}")
        if (user2[1] == 'i'):
            data_dic['i'] = f"{CYAN}{user2[0]}{RESET}"
            board(sign3=f"{YELLOW} <<--- {RESET}")

    Choice_dict['Player 2'] = user2[0]
    winner = check(data_dic['a'],data_dic['b'],data_dic['c'],data_dic['d'],data_dic['e'],data_dic['f']
        ,data_dic['g'],data_dic['h'],data_dic['i'])
    if (winner != " "):
        continuous_check = (True,winner)
```

```python
                break


        counter += 1



if f == 'a' or f == 'b':
    pass
else:
    if (continuous_check[0] == True):
        key_list = list(Choice_dict.keys())
        value_list = list(Choice_dict.values())
        if (choices == 2):
            position1 = value_list.index('X')
            position2 = value_list.index(var)
        else:
            position1 = value_list.index('X')
            position2 = value_list.index('O')
        if (continuous_check[1] == f'{RED}X{RESET}'):
            if choices == 2:
                if (key_list[position1] == "Player 1"):
                    print(f"{MAGENTA}Congratulations! Winner of the Game is{BOLD} {name1}")
                else:
                    print(f"{MAGENTA}Congratulations! Winner of the Game is{BOLD} Computer.")
            else:
                if (key_list[position1] == "Player 1"):
                    print(f"{MAGENTA}Congratulations! Winner of the Game is{BOLD} {name1}")
                else:
                    print(f"{MAGENTA}Congratulations! Winner of the Game is{BOLD} {name2}")
        else:
            if choices == 2:
                if (key_list[position2] == "Player 1"):
                    print(f"{MAGENTA}Congratulations! Winner of the Game is{BOLD} {name1}")
                else:
```

```python
                    print(f"{MAGENTA}Congratulations! Winner of the Game is{BOLD} Computer.")
                else:
                    if (key_list[position2] == "Player 1"):
                        print(f"{MAGENTA}Congratulations! Winner of the Game is{BOLD} {name1}")
                    else:
                        print(f"{MAGENTA}Congratulations! Winner of the Game is{BOLD} {name2}")


        else:
            pass
if __name__ == '__main__':
    game()



"""
My Approach is that I will create  a dictionary and each time a column is filed I will pass the column name and the value given to that column
in the dictionary and while giving input I will check if that column is already filled and also the checking will be done using the dictionary.
Also the Colors of each Variable and the Arrows pointing to the current input column is given, the Warning situation such as If entered a column
already filled then it will give warning and choice to re-enter and if Choosed a variable which is not the same as choosed in the beginning
it will give a Warning and choice to re-enter, and all the Warnings and choice to re-enter are given untill the Choice is True with the given
Conditions,
as mentioned before.The Player who takes the winning variable, his name gets displayed at the end while declearing the winner, and this is acheived
by storing the variable choosed by each user at the beginning and then calling the key of the dictionary from the value.
```

TESTING

Software Testing is an empirical investigation conducted to provide stakeholders with information about the quality of the product or service under test[1] , with respect to the context in which it is intended to operate. Software Testing also provides an objective, independent view of the software to allow the business to appreciate and understand the risks at implementation of the software. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs.

It can also be stated as the process of validating and verifying that a software program/application/product meets the business and technical requirements that guided its design and development, so that it works as expected and can be implemented with the same characteristics. Software Testing, depending on the testing method employed, can be implemented at any time in the development process, however the most test effort is employed after the requirements have been defined and coding process has been completed.

TESTING METHODS

Software testing methods are traditionally divided into black box testing and white box testing. These two approaches are used to describe the point of view that a test engineer takes when designing test cases.

BLACK BOX TESTING

Black box testing treats the software as a "black box," without any knowledge of internal implementation. Black box testing methods include: equivalence partitioning, boundary value analysis, all-pairs testing, fuzz testing, model-based testing, traceability matrix, exploratory testing and specification-based testing.

SPECIFICATION-BASED TESTING

Specification-based testing aims to test the functionality of software according to the applicable requirements.[16] Thus, the tester inputs data into, and only sees

the output from, the test object. This level of testing usually requires thorough test cases to be provided to the tester, who then can simply verify that for a given input, the output value (or behaviour), either "is" or "is not" the same as the expected value specified in the test case. Specification-based testing is necessary, but it is insufficient to guard against certain risks

ADVANTAGES AND DISADVANTAGES

The black box tester has no "bonds" with the code, and a tester's perception is very simple: a code must have bugs. Using the principle, "Ask and you shall receive," black box testers find bugs where programmers don't. But, on the other hand, black box testing has been said to be "like a walk in a dark labyrinth without a flashlight," because the tester doesn't know how the software being tested was actually constructed.

That's why there are situations when (1) a black box tester writes many test cases to check something that can be tested by only one test case, and/or (2) some parts of the back end are not tested at all. Therefore, black box testing has the advantage of "an unaffiliated opinion," on the one hand, and the disadvantage of "blind exploring," on the other.

WHITE BOX TESTING

White box testing, by contrast to black box testing, is when the tester has access to the internal data structures and algorithms (and the code that implement these)

Types of white box testing:-

The following types of white box testing exist:

- api testing - Testing of the application using Public and Private APIs.
- Code coverage - creating tests to satisfy some criteria of code coverage.

For example, the test designer can create tests to cause all statements in the program to be executed at least once.

- fault injection methods.

- mutation testing methods.
- static testing - White box testing includes all static testing.

CODE COMPLETENESS EVALUATION

White box testing methods can also be used to evaluate the completeness of a test suite that was created with black box testing methods. This allows the software team to examine parts of a system that are rarely tested and ensures that the most important function points have been tested.

Two common forms of code coverage are:
- Function Coverage: Which reports on functions executed and
- Statement Coverage: Which reports on the number of lines executed to complete the test.

They both return coverage metric, measured as a percentage

25

HARDWARE AND SOFTWARE REQUIREMENTS

I.OPERATING SYSTEM : WINDOWS 7 AND ABOVE

II. PROCESSOR : PENTIUM(ANY) OR AMD

ATHALON(3800+- 4200+ DUAL CORE)

III. MOTHERBOARD : 1.845 OR 915,995 FOR PENTIUM 0R MSI

K9MM-V VIA K8M800+8237R PLUS

CHIPSET FOR AMD ATHALON

IV. RAM : 512MB+

V. Hard disk : SATA 40 GB OR ABOVE

VI. CD/DVD r/w multi drive combo: (If back up required)

VII. FLOPPY DRIVE 1.44 MB : (If Backup required)

VIII. MONITOR 14.1 or 15 -17 inch

IX. Key board and mouse

X. Printer : (if print is required – [Hard copy])

SOFTWARE REQUIREMENTS:

I. Windows OS
II. Python

.

BIBLIOGRAPHY

1. *Computer science With Python - Class XI By : Sumita Arora A*
2. *Project Report On Blood Bank Management System (BBMS) By : Praveen M Jigajinni*
3. *Website:* **https://www.w3resource.com**
4. *Website:***https:// https://github.com**

***