VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.



LAB REPORT on

ARTIFICIAL INTELLIGENCE

Submitted by

KEERTHI P REDDY (1BM21CS090)

in partial fulfillment for the award of the degree of BACHELOR OF ENGINEERING in COMPUTER SCIENCE AND ENGINEERING



B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
Nov-2023 to Feb-2024

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering



CERTIFICATE

This is to certify that the Lab work entitled "ARTIFICIAL INTELLIGENCE" carried out by KEERTHI P REDDY (1BM21CS090), who is bonafide student of B.M.S. College of Engineering. It is in partial fulfillment for the award of Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belgaum during the academic semester June-2023 to Sep-2023. The Lab report has been approved as it satisfies the academic requirements in respect of a ARTIFICIAL INTELLIGENCE LAB (22CS5PCAIN) work prescribed for the said degree.

Dr. Asha G R Dr. Jyothi S Nayak

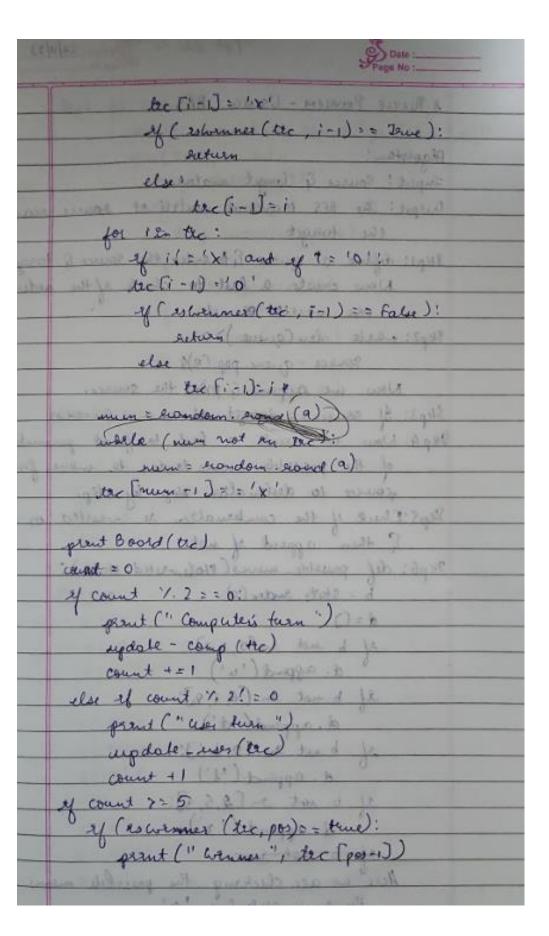
Assistant Professor Professor and Head

Department of CSE Department of CSE

BMSCE, Bengaluru BMSCE, Bengaluru

Program 1: Tic Tac Toe

Tic Tac Toe
board [' for en range()] this emport random tre =[1,2,3,4,5,6,1,8,9] def part Board (tre): part (tre[0] +'1' + tre[1] + '1' + tre[2]) part ("") part (tre[5] + '1' + tre[4] + '1' + tre[5]) part (tre[6] + '1' + tre[4] + '1' + tre[8]) def ashrumes(tre, pos): of tre[0] = tre[6] and tre[4] == tre[6] return true else of tre[pass[3+1] = tre[pass[3+2] and tre[3+2] = : tre[pass[3+3]: part (3+2] = : tre[pass[3+3]: part (3+2] = : tre[pass[3+3]:
support examples tre =[1,2,34,5,6,1,8,9] def posset Boord (tre): psent (tre[o] +'1'+ tre[1]+'1'+ tre[2]) prent ("") psent (tre[s]+'1'+ tre[4]+'1'+ tre[5]) psent ("") psent (tre[s]+'1'+ tre[4]+'1'+ tre[5]) def exhammes (tre, pos): y tre[o] = tre[s] and tre[s] = tre[s] = stre [2] = : tre[4] and tre[s] = tre[6] return true ulse of tre[pos[3+1] = tre[pos[3+2] and tre (3+2] = : tre[pos[3+3]: peturn true
support examples tre =[1,2,34,5,6,1,8,9] def posset Boord (tre): psent (tre[o] +'1'+ tre[1]+'1'+ tre[2]) prent ("") psent (tre[s]+'1'+ tre[4]+'1'+ tre[5]) psent ("") psent (tre[s]+'1'+ tre[4]+'1'+ tre[5]) def exhammes (tre, pos): y tre[o] = tre[s] and tre[s] = tre[s] = stre [2] = : tre[4] and tre[s] = tre[6] return true ulse of tre[pos[3+1] = tre[pos[3+2] and tre (3+2] = : tre[pos[3+3]: peturn true
the =[1,2,3,4,5,6,1,8,9] def point Boord (tec): pernt (the[0]+'1'+ tre[1]+'1'+ tre[2]) print ("") pernt (tre[5]+'1'+ tre[4]+'1'+ tre[5]) pernt (tre[5]+'1'+ tre[4]+'1'+ tre[5]) pernt (tre[6]+'1'+ tre[7]+'1'+ tre[5]) def ashermes (tre, pos): y tre[0]=tie[6] and tre[4]== tre[8] or tre[2]==tre[4] and tre[4]== tre[6]: return true ulse of tre[pos[13+1]=tre[pos[13+2]) and tre[3+2]== tre[pos[13+3]: peturn true
the =[1,2,3,4,5,6,1,8,9] def passet Boord (the): passet (the[0] +'1' + the [1]+'1' + the [2]) passet ("") passet ("") passet ("") passet ("") passet ("") passet (the [6] +'1' + the [7] +'1' + the [8]) def ashermes (the, pos): of the [0] = the [6] and the [4] = : the [8] of athe [2] = : the [4] and the [4] = : the [6]: return true also of the [pos[13+1] = the [pos[13+2] and the [3+2] = : the [pos[13+3]: pasture the
def part Boord (tic): part (tic[0] +'1' + tic[1] + '1' + tic[2]) print ("") print (tic[5] + '1' + tic[4] + '1' + tic[5]) pent ("") pent (tic[6] + '1' + tic[7] + '1' + tic[8]) def ashernes (tic, pos): If tic[0] = tic[6] and tic[4] = : tic[8] & etic[2] = : tic[4] and tic[4] = : tic[6]. return time ulse if tic[pos[13+1] = tic[pos[13+2] and tic[3+2] = : tic[pos[13+3]: exturn time
prent (tre[0] +'1' + tre[1]+'1' + tre[2]) prent ("") prent (tre[5]+'1' + tre[4]+'1' + tre[5]) prent ("") prent (tre[6]+'1' + tre[7]+'1' + tre[8]) def exhannes (tre, pos): y tre[0] = tre[4] and tre[4] = tre[8] or stre[2] = : tre[4] and tre[4] = : tre[6]: return true else of tre[pos[13+1] = tre[pos 113+3]: enturn true enturn true
pent (tec[5]+'1'+ tre[4]+'1'+ tre[5]) pent ("") pent (tec[6]+'1'+tre[7]+'1'+ tre[8]) def estermes (tre, pos): y tre (0]= tie [i] and tre [4]:= tre[8] or tre (2):= tre[4] and tre[4]:= tre[6]: return true ulse of tre (pos[3+1]:= tre[pos[13+2] and tre (3+2]:= tre [pos 113+3]: enturn true
pent ("") pent (tic [6] + '1' + tic [7] + '1' + tic [8]) def ashermer (trc, pos): y trc [0] = tic [i] and trc [4] = : trc [8] or strc [2] = : trc [4] and trc [4] = : tre [6]: return true ulse of trc [pos[13+1] = trc [pos 113+2] and bac [3+2] = : trc [pos 113+3]: seturn true
pent (" ") pent (tic [6] + '1' + tic [7] + '1' + tic [8]) def ashermer (trc, pos): If trc [0] = tic [i] and trc [4] = : trc [8] or strc [2] = : trc [4] and trc [4] = : tre [6]. return true ulse of trc [pos[3+1] = trc [pos 113+2] and bac (3+2] = : trc [pos 113+3]: seturn true
def esternmes (trc, pos): If trc[0] = tic[4] and trc[4] = : trc[8] or strc[2] = : trc[4] and trc[4] = : trc[6]. return true ulse of trc[pos[3+1] = trc[pos[13+2] and trc[3+2] = : trc[pos[13+3]: preturn true
y tre (0] = tic [i] and tre (4) = : tre (8) or tre (2) = : tre [4] and tre [4] = : tre [6]: return true ulse of tre [pos[13+1] = tre [pos 113+2] and tre (3+2] = : tre [pos 113+3]: seturn true
y tre (o] = tic [i] and tre [4] = : tre [8] or stre (2) = : tre [4] and tre [4] = : tre [6]: return true else of tre [pos[13+1] = tre [pos[13+2] and tre (3+2] = : tre [pos 113+3]: return true
ster (2) = = tec [4] and tec [4] = = tec [6]. return true else of tec [pos[13+1] = tre[pos[13+2] and tec [3+2] = = tre [pos 113+3]: seturn true
else of exc[pos[13+1]: tre[pos[13+2] and tre (3+2] = : tre [pos 113+3]: neturn true
seturn true
seturn true
return false
V V
del lando la vez (tro)
def update-user (tre):
while (num not en tec):
water and and the second of th
tec [num - 1] = = "0" Fake a no, on the box
yee (was
def update - comp (tie):
for i an tre:
for i an tre: of i! = 'x' and i! = 'o':



KEERTHI P REDDY 1BM21CS090

[1, 2, 3, 4, 5, 6, 7, 8, 9]

i		
1	2	3
4	5	 6
7	 8	9

computer's	turn :	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
+		+
1	2	3
+	 	 +
	_	
4	5	X
÷		
7	8	9
		i į
+		

Your turn :

enter a number on the board :1

 0 	2	 3 			
0	Х	x			
7	8	 9 			
computer's turn :					

computer's turn :

0	2	3
0	х	x
X	8	9

Your turn :

enter a number on the board :2

0	2	3
4	5	x
7	8	9

computer's turn :

compacer 5 carri .				
0	 2 	 3 		
 4 	 x 	x		
7	 8 	9		

Your turn :

enter a number on the board :4

0	0	3
0	Х	x
x	8	9

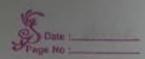
computer's turn :

4		
0	 0 	
0	 x 	x
X	 8 	9

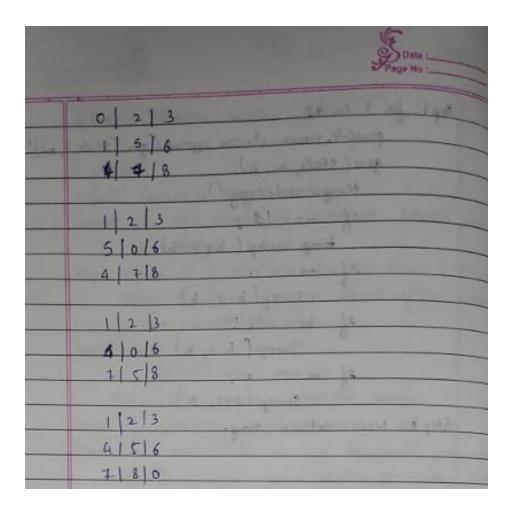
winner is X

Program 2: 8 Puzzle Problem Using BFS

	4/A-24-11-23 Mate 24/
& PUZZLE PROBLEM .	
Mary Miller	and the same of the same of the
Algorithm:	
Junet! Come &	Jarget motors
Output : The BES	traveral until At source
the tange	traveral until et source
Step 1: defens a	The E to cutor the come of
New creat	e a last to check of the
are ve	ested or not.
Skp2: onhela den	
	quene pap (0)/4
Now we a	uppend & prent the source.
exps. 3/ source =	= toract as a -
The same	cheate a last to land
	alderation ston a olo
IN	Laurenada 11 11111
	MANON 31 HOL
- Francop	- mines (state insited of 1).
stote = d	suder (0)
d = []	and bellevent the second
y b not	2m (0, 1, 2):
d. ap	pand ('u')
of b not	m (6, 1, 8):
d.aj	ppend ('d')
of bust	en [0,3,6]:
d.0	append ('1')
af b ist	t on [2,5,0]:
	ppend ('r')
2 6 xx	
Here we are	checking the possible mor
the empty	Stale, here 'o'



	Spage No :
	94cp 1: fes 9 land:
	posselele. mones. et. con. append (gan, (stale, i, b)):
	gen(state, m, b):
	temp = State copy ()
Ī	ef m >> (d';
	temp swap (b+3, b)
	of m== 'ce':
	Swap (b-3, b)
	2/ m=='1':
	Swap (b-1, b)
	2/ m = = 'x'!
	3crap (b+1, b)
	Step 8: Now return temp.
	31214
	8 + 4 0 1 2
	3 1 2 1 3 4 5
Ī	0 5 6 6 7 8
	Svc +gt
	OUTPUT:
	1 2 3
	A 5 6
	0 7 18
	1 2 3
	0 5 6
	4 7 8
	1 2 3
	41516
	7/0/8



1 2 3	3			
4 5 6	5	1	2	3
0 7 8	3	5	0	6
	-			
1 2 3	3	4	7	8
0 5 6	5			
4 7 8	3	1	2	3
	-	4	0	6
1 2 3	3	7	5	8
4 5 6	5			
7 0 8	3	1	2	3
	-			
0 2 3	3	4	5	6
	5	7	8	0
4 7 8	3			
	-	Suc	cess	5

Program 3: 8 Puzzle Problem using Iterative Deepening Search algorithm

	MA81,2/23 Space 8/12/23
	8 puzzk stoustene deepening worch algoretum.
	Algorethm:
	Skept: Inetaloge => enetial state = [] & goal of the
1	gool state = [1, 2, 3, 4, 5, 6, 7, 0]
	# 0 is the blank space.
	Expand snetal state
	depth-limeted-sexich (depth) as performed.
	ef rest. state > goal
	alse for
	alse for meghbour in get-weighbour
	(node state) child = puzzle node (næghbous, nodé)
	result = depth-limsted scorch (depth-1)
-	If result => True;
	Ste. 3: Alles as also also also also
	Step 3: After one ateralian where depth = 1, depth + +
	depth laweted - search as performed ago
	step 4: Here get neighbour will generate the
	This is done by swopping the blank
_	
2	the other of reached the goal state.
	the other greatest the good state.
	1 2 4 6 5 6 4 6 5 6 4 6 5 6 6 6 6 6 6 6 6 6
	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

KEERTHI P REDDY 1BM21CS090

Success!! It is possible to solve 8 Puzzle problem
Path: [[1, 2, 3, 4, 5, 6, 0, 7, 8], [1, 2, 3, 4, 5, 6, 7, 0, 8], [1, 2, 3, 4, 5, 6, 7, 8, 0]]

Program 4: 8 Puzzle Problem using A*

12	Fege No. 2 (2/ 23
	Solveng 8-puzzle wang A* algoritam
	Algorathm:
and the	Step : Inetial state & goal state are expoted.
1000	Here we take a count of humsters
	en each step.
	fralue = h value + pathicost
42.24	Step2: Frestly expand the node. In
	Find the location of empty lese
400	Now togenerate the heuristic value use the
	function. f(x). h(x)+g(x)
	In(x) => mess placed tiles
	g(x) => depth from storling node (parti cost)
	Step3: Two lests are maintained open & close
	The open last stores the moder(states) general
	Sort using f(x) values.
-	The close last stones using emploring of
	nodes 4 summed from open.
	orb The state is searched when 1/8/20
	Thro implies that all the tiles are is
0	correct paretton.
d	R = 1123
to de	180 8 4 5 6 N=2
10	0 7 8
40	
	1 2 3
	4 5 6 451 0 5 6 623
	103
	1 5 6 A=0
	780
The same of the sa	

\'/

```
Enter the start state matrix
1 2 3
4 5 6
_ 7 8
Enter the goal state matrix
1 2 3
4 5 6
78_
 \'/
1 2 3
4 5 6
_ 7 8
\'/
1 2 3
```

Program 5: Vacuum Cleaner Agent

	Vaccum Cleans Agent
	varcum cusars of
	Algorithm
	Skept: Instralize the storting state & goal et
	The goal state is to clean the swom
	A Y B.
	Step 2: If status = Dirty then clean
	else of location = H 4 status = Crean the
	exturn ærglit
-34	else of location = 0 4 status = clown the
-	extrem left
	else enet.
Chas	Step 3: If both the localean are clean
	then ext. A B A B
30	
0	A 3
566	The same building as their last Discount
	The section that are the college

KEERTHI P REDDY 1BM21CS090

0 indicates clean and 1 indicates dirty Enter Location of Vacuum Enter status of al Enter status of other room1 Vacuum is placed in location B Location B is Dirty. COST for CLEANING 1 Location B has been Cleaned. Location A is Dirty. Moving LEFT to the Location A. COST for moving LEFT2 COST for SUCK 3 Location A has been Cleaned. GOAL STATE: {'A': '0', 'B': '0'} Performance Measurement: 3

Program 6: Knowledge Base Entailment

	lRA29-12-23 \$ Date: 29/12/21
	Knowledge Base Entaelment.
	Algorotum:
3/10	SATISFIABLE SAME OR, AND, NOT, IMPLIES,
	Itep2: Cueste knowledgebase using variables
1,0%	Step 3: Nour define the knowledge base
	Implies (p, q),
	Implies (q, x), Not (x)
	Then return knowledge-base
	Step 4: Sefrue the query entorles entarluent: Satesfrable (And (knowledge base, Not Ge
Sec	Step 5: of name = " marn-":
	kb = create - knowledge - base ()
	Step 6: Defrue query
	Step 7: Check of the query enthals kb
	Stop & Print Results.

KEERTHI P REDDY 1BM21CS090

Knowledge Base: $\sim r & (Implies(p, q)) & (Implies(q, r))$

Query: p Query entails Knowledge Base: False

Program 7: Knowledge Base Resolution

	Page No:
knowledge Base Resolution.	nothing all
Algorithm:	1 = Almonto
Step 1: Create knowledge	base using some
Step 2: Choose what you	need to check
Step 3: Now negate the needs to entalled	vorable which
mondage base.	COT CO.
stepti: Now of you go	t the output then
Step 5: Now Prent resul	b.

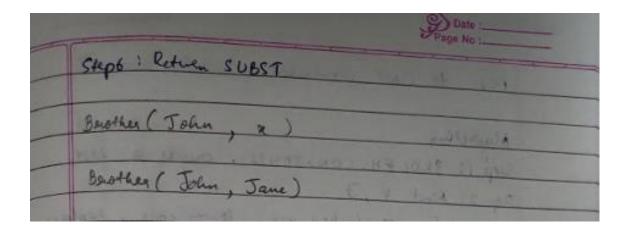
Step	Clause	Derivation
1.	R∨~P	Given.
2.	R∨~Q	Given.
3.	~RvP	Given.
4.	~RvQ	Given.
5.	~R	Negated conclusion.
6.		Resolved Rv~P and ~RvP to Rv~R, which is in turn null.
A contr	radiction	is found when aR is assumed as true. Hence R is true

```
Step | Clause | Derivation
     | PvQ | Given.
2.
      | ∼PvR | Given.
3.
      | ~Q∨R | Given.
4.
      ~R
              | Negated conclusion.
      | QvR | Resolved from PvQ and ~PvR.
5.
      PvR
             Resolved from PvQ and ~QvR.
6.
              Resolved from ~PvR and ~R.
7.
      ~P
              Resolved from ~QvR and ~R.
      | ~Q
8.
              Resolved from ~R and QvR.
       | Q
9.
10.
       l P
              Resolved from ~R and PvR.
11.
              Resolved from QvR and ~Q.
             Resolved R and ~R to Rv~R, which is in turn null.
12.
A contradiction is found when ~R is assumed as true. Hence, R is true.
```

Step	Clause	Derivation
1.	PvQ	Given.
2.	PvR	Given.
3.	~PvR	Given.
4.	RvS	Given.
5.	R∨~Q	Given.
6.	~Sv~Q	Given.
7.	∼R	Negated conclusion.
8.	QvR	Resolved from PvQ and ~PvR.
9.	Pv~S	Resolved from PvQ and ~Sv~Q.
10.	P	Resolved from PvR and ~R.
11.	~P	Resolved from ~PvR and ~R.
12.	R∨~S	Resolved from ~PvR and Pv~S.
13.	R	Resolved from ~PvR and P.
14.	S	Resolved from RvS and ~R.
15.	~Q	Resolved from Rv~Q and ~R.
16.	Q	Resolved from ~R and QvR.
17.	~S	Resolved from ~R and Rv~S.
18.	I	Resolved ~R and R to ~RvR, which is in turn null.
A contr	radiction	is found when ~R is assumed as true. Hence, R is true.

Program 8: Unification

	PA19-1-23 Spage No: 19/01/23
	Unification
	Algorithm:
	Step 1: If term 1 or term 2 is a wareal or
	Constant then:
	of term 1 or term 2 are adentical
	Action 10.1
	else ef term 1 es avaelable
- 10	If term 1 occurs in term 2
	return fael
	else
	return fort (terms /term)}
	the if term 2 is a variable
	etuen Fall
	extrem Fall
	system fael
	Step 2: If predicate (term 1) \$ predicate (term 2)
	return fall
	Step 3: Number of assignments are not equal
	suturn fall
	step 4: set (subste) to VII
	8tcp 5: For i=1 to the no. of elements on team!
	Call engly (ith team 1, ith term 2)
	put nesult into 5
	S 2 fael
	Return fael
	af S+NIL
	appliedy S to sumander of both a g. L.
	apply S to sumander of both (1 Ext. SUBST = Append CS, SUBST)



```
Substitutions:
[('X', 'Richard')]
Substitutions:
[('A', 'y'), ('mother(y)', 'x')]
```

Program 9: FOL to CNF conversion

	9 Oate :
FOL	to CNF conversion
Alge	rething:
Step	1: SKOL FH_CONSTANTS, create a list
Step	2: Food \(\Pi \), \(\Pi \)
	ef attrebutes are lower cose, explace
	them with skolem constant
	sumpose used skolem constant or funct
	from the lest
	If the attributes are both lower cos
	appearance attribute with a
	skolem function.
Step 3	Replace <= 1 - 1
	trongoin - as Q = (P => Q) ~ (Q => P
Step 4:	enplace => wrth '-'
Steps	Apply demorgani law
	euplace - [
	seplece I [and of (1 was present)
	zuprace 1
	as TPITA of Ca was prom
	rugue i wan
Step 6:	Show result.
-	
V 24	keng(n)=> Person(n)

Program 10: Forward Chaining

	Sonta 34/01 /24
Forward Charning	
Algorithm;	
Step1: Input knowledge bo	8 .
Step1: Input knowledge base Step2: for i an kB	4 query
af i = = query	
sufuen ferre	
ef ' => ' en i	
Splat the and	the said
Splet the and,	rus past
add she to kB	
Step 3. To remove warrables	
of i lower ()	
epplace the vorsa	ale with courtst
Fg: kB?	
leng (x) & gereedy (2) =) evil(x)
greedy (John)	
king Richard	
0 /	
query:	
query:	

KEERTHI P REDDY 1BM21CS090

Querying criminal(x):

1. criminal(West)

All facts:

- 1. criminal(West)
- enemy(Nono,America)
- 3. sells(West,M1,Nono)
- 4. american(West)
- 5. missile(M1)
- 6. hostile(Nono)
- 7. weapon(M1)
- 8. owns(Nono,M1)

Querying evil(x):

- evil(Richard)
- 2. evil(John)