**Supplementary Material for Session 4**

1. **Major Components of Prolog Code for Hill-Climbing Local Search and Genetic Algorithms:**

**% Code for generating the successors of a 8-queens' state given as a list of 8 digits**

gnrt\_sucsr(L):- assert(id(1)), assert(state(1, 'c', L, 50)),

incr\_id, mk\_new(1, L), retract(id(\_)), evaluate.

incr\_id:-id(V), V1 is V+1, retract(id(\_)), assert(id(V1)).

mk\_new(9, \_):-!.

mk\_new(N, L):- nthel(N, L, X), del\_el(X, [1,2,3,4,5,6,7,8], L1),

cng\_mk(N, L, L1), N1 is N+1, mk\_new(N1, L).

cng\_mk(\_, \_, []):-!.

cng\_mk(N, L, L1):- L1=[H|T], rplc\_nthel(N, H, L, L2), id(Id), /\* id/1 is a dynamic data \*/

assert(state(Id, 's', L2, 50)), incr\_id, cng\_mk(N, L, T).

**% Code for determination and display of the best state**

checkall:- state(\_, 'c', \_, V1), threshold(V2), V1 >= V2, I is 1, dsply(I), !.

checkall:- best(I1,V1), threshold(V2), V1 >= V2, I is I1, dsply(I), !.

checkall:- state(\_, 'c', \_, V1), best(I, V2) ,V2>V1,state(I, \_, L, \_),

retractall(state(\_, \_, \_, \_)),write\_list(['\nIteration max: ', V2]),

gnrt\_sucsr(L), !.

checkall:- restrt, !.

best(I, Max):- state(\_, 's', \_, Val), assert(max\_val(Val)),

updt\_max, max\_val(Max), state(I, \_, \_, Max), retract(max\_val(\_)), !.

updt\_max:- state(\_, \_, \_, V2), max\_val(V1), V2>V1,

retract(max\_val(\_)), assert(max\_val(V2)), fail.

updt\_max:-!.

**% Performing Crossover**

go\_cross(X,Y,CP):- state(X,'p',L1,\_), state(Y,'p',L2,\_),CP1 is 8-CP,

del\_1st\_n\_el(L1,CP,L12),del\_last\_n\_el(L1,CP1,L11),

del\_1st\_n\_el(L2,CP,L22),del\_last\_n\_el(L2,CP1,L21),

append(L11,L22,LO1),append(L21,L12,LO2), count\_sts(\_,N),

N1 is N+1, N2 is N+2,

assert(state(N1,'o',LO1,50)), assert(state(N2,'o',LO2,50)).

**% Performing Mutation**

do\_mutn:- count\_sts('o',N), N1 is random(N)+1,

assert(id1(0)),get\_offspr(N1,I,T,L,V), retract(id1(\_)),

N2 is random(8)+1, N3 is random(8)+1, rplc\_nthel(N2,N3,L,L1),

retract(state(I,T,L,V)), assert(state(I,T,L1,50)).

get\_offspr(N1,I,'o',L,V):- state(I,'o',L,V),incr\_id1, id1(N), N1=N,!.

1. **Sample Codes for Object Oriented Programming with Python**
2. **Simple example of Class, Objects and Inheritance**

class Animal:

def \_\_init\_\_(self, m):

self.movement = m

def printAnimal(self):

print("Movement: "+self.movement)

class Mammal(Animal):

def \_\_init\_\_(mml, wb, m):

Animal.\_\_init\_\_(mml, m)

mml.warm\_blooded= wb

def printMammal(self):

self.printAnimal()

print("Warm blooded: "+self.warm\_blooded)

class Cat(Mammal):

def \_\_init\_\_(kt, c, nol, wb, m):

Mammal.\_\_init\_\_(kt, wb, m)

kt.color = c

kt.no\_of\_legs= nol

def printCat(kt):

kt.printMammal()

print("Color: "+kt.color+"\n"+

"Number of Legs: "+str(kt.no\_of\_legs))

C1=Cat("White", 4, "Yes", "Yes")

print("\nDetailed Information of Cat:\n")

C1.printCat()

M1=Mammal("Yes", "Yes")

print("\nDetailed Information of Mammal:\n")

M1.printMammal()

1. **Another simple example with Inheritance and Overloading**

class Calculation:

def calcVolume(self,arg1,arg2=None, arg3=None):

if arg2 != None:

return arg1\*arg2\*arg3

else:

return 4\*3.14\*arg1\*\*3/3

class Sphere(Calculation):

def \_\_init\_\_(sphr, r):

sphr.baseRadius = r

def displaySphere(self):

print("Sphere volume: ", end='')

print(self.calcVolume(self.baseRadius))

class Cube(Calculation):

def \_\_init\_\_(cb, l, w, h):

cb.length = l

cb.width = w

cb.height = h

def displayCube(c):

print("Cube volume: ", end='')

print (c.calcVolume(c.length,

c.width,c.height))

S1=Sphere(float(input("\nSpere Radious:")))

print("\nSphere Volume Calculation")

S1.displaySphere()

C1=Cube(float(input("\nCube length:")),float(input("Cube width:")),

float(input("Cube height:")))

print("\nCube Volume Calculation")

C1.displayCube()

1. **Sample Code for a Rule based System in Prolog**

**% Rules:**

hypothesis(Patient, flu):-

symptom(Patient, headache), symptom(Patient, fever),

symptom(Patient, runny\_nose).

hypothesis(Patient, common\_cold):-

symptom(Patient, sneezing),

symptom(Patient, runny\_nose).

**% Facts / Data:**

symptom('Rahim', headache).

symptom('Karim', headache).

symptom('Hasib', headache).

symptom('Karim', fever).

symptom('Hasib', fever).

symptom('Hasib', sneezing).

symptom('Rahim', sneezing).

symptom('Karim', runny\_nose).

symptom('Rahim', runny\_nose).