Siddle orderick

alum lack (cope) of the (constant) and all (re gradulate) for g is emorthed) and all (re gradulate) cher ke: cher ke: cher fouts - subte) subt (coupt e): che the (cope e): che the (cope e): che the (cope e): for im subt (mylicularies and (Implication (e))) the: subt implicularies and (Implication (e)) if and in subt implication (e) print (subt) (subt) for fout (subt) print (subt) predicate = fact (e) producate: print (subt) print (subt) for i, g in comments (set ([5] impuliaries for for for e); print (subt facts:) for i, g in comments (set ([5] impuliaries for for for e); print (subt) (set ([5] in [5]); print (subt) (set ([5] in [5]);			papergrid
for f is two this) and all (IF get Savieties der the. alf mid (ut): suf fouts: sutco suf implications ut() def the (up, e): suf implications and (Implication (e)) for im suf implications me = i treatment (uf fouts) for im facts: for facts: from facts: i fout(5) produced == fact (e), produced: print (5' aurying 5 cd:) for i facts: def display (suf) print ("Mi (uto:") for i, 5 in cumunous (set (15. impunion for 5 m.) ent fouts):		18 predicate 18 attribute "	
def mid (wet): Suf feuts suls) Suf feuts suls) Las: Suf implications add (Implication (c)) Las: Suf implications add (Implication (c)) Las: Suf feuts add (suls) Jaca: Suf feuts industr (wef feuts) Jaca: Suf feuts indefeuts Jaca: Jaca		Edim fort (robar) of miles	
class to: alf wit (wt): Suf feuts: sut() det ten (wf, e): Las: Suf implications and (Implication (e)) des: Suf implications and (Implication (e)) for im Suf, implications The i mathest (wf, feuts) grave: Suf feuts indd(su) dif gury (suf, e): facts = sut Po ([fr. repression for f in suf feuts]) is 1 print (f aurying fed:) if fact (d) predicate == fact (e), producate: print(f Vifit, fel!) dif display (suf) print (M feuts:") for i, f in auruments (set ([f. repression for f in suf feats)):		Son & in comment and by	Lite of Sames
def wit (wet): Suf, feuts: sut() def tue (wet, e): J'=>' in e: Suf, facts, and (Fact (e)) dae: Suf, facts, and (Fact (e)) for im Suf, implications wh = i waluate (wef, facts) if suc suf, facts, and (Fact) facts = set Pro ([fe, commion for f in suf, facts]) print (f' Ownging & ch: ') for in facts: if Fact (s) product = = fact (e), product : print (f' Out (d)) product = = fact (e), product : print (f' Utility, \$55') duf display (suf) print (fut facts: ') for i, S in comminate (set (if comminate for f in ent)):	<u>#</u>		nent
suff implications est() def test (suff, e):		선택하게 하는 사람들은 그 이번 문화를 모르는 하다. 그들은 사람들은 사람들이 되어 보고 있는 것이 되는 것이라면 하는 것이다. 이번 것은 그렇게 되었다는 것이 없어요?	
suff implications with dif test (set, e): dif test (set, e): suff implications, add (Implication (e)) else: suff implications alse: suff facts add (fact (e)) for im suff, implications who = it walests (set, facts) def query (suff, e): facts = set Pro ([fr. corporation for f in set, facts]) i=1 print (f' aurying & ch:') for f in facts: if fact(s) predicate == fact (c) producate: print(f' 'third, sgs') def display (setf) print c'ru (suff):') for i, f in comments (set ([s. corporation for f in set, facts)):		olf-init (suf):	
self. implications est () def ter (seef, e): >' me: self. facts. ash (fact (e)) for im Self. melications ord = i tendents (self. facts) if sec: self. facts. add(sec) def group (self. e): facts = self 90 ([fr. copration for f in self. facts]) i=1 print (f' aurying & ch:') for f mid (self) print ('He facts') for i, f in commonts (set ([f. copration for f in self. facts)): est. facts)):		Suf , feets = self)	
def test (seef, e): 21 - 2' me: Seef: implications, add (Implication (e)) stor: Set facts. add (Fact (e)) fore im self, implications The implications		sulf. implication- utc	
suf implications, odd (Implication (e)) else: Suf puts at (fact (e)) for im Suf, implication rest i traduate (suf facts) y res suff, facts, odd (res) def gruny (suf, e): facts = sut Po ([fr. copration for f in ent facts]) i=1 print (5' Quenying & (3:') for f in facts: y fact (3) predicate == fact (e), predicate: print(f'(this. 15')) def display (suff) print ("Mu facts.") for i, f in commercian (set ([1. copration for f in ent. 1)):	1 T		
suf implications, odd (Implication (e)) else: Suf facts. add (Fact (e)) for im Suf, implication rect i traduata (suf facts) y rec. suff, facts, odd (rec) def group (suf, e): facts = sut Po ([fr. copration for f in ent facts]) i=1 print (5' Ourning & cd: ') for f in facts: y fact (d) predicate == fact (e), predicate: print(f'\tail_i, \forall_i) i+=1 def display (suf) print ("Mu facts:") for i, f in commercia (set ([fr. copration for f in ent facts)):		def tur (wf, e):	
set futs at (Fart (a)) for in set, implications act = i teatuate (set, facts) if his set gury (set, e): facts = set Pro ([fr. coparison for f in set, facts]) i=1 print (5' Quenying & c3: ') for f in facts: print(5' \text{ \text{ \text{producate}} == fact (e), producate: print(5' \text{ \text{ \text{producate}} = fact (e), producate: print(5' \text{ \text{ \text{ \text{ \text{ \text{print}}}}}) def display (sect) print ("He facts") for i, 5 in commence (set (15, copacison for 5 m) set, facts")):		il '=>' me:	
set futs at (Fart (a)) for in set, implications act = i teatuate (set, facts) if his set gury (set, e): facts = set Pro ([fr. coparison for f in set, facts]) i=1 print (5' Quenying & c3: ') for f in facts: print(5' \text{ \text{ \text{producate}} == fact (e), producate: print(5' \text{ \text{ \text{producate}} = fact (e), producate: print(5' \text{ \text{ \text{ \text{ \text{ \text{print}}}}}) def display (sect) print ("He facts") for i, 5 in commence (set (15, copacison for 5 m) set, facts")):		suf implications, add (7, 1:	
for in Self, implications ors = i traduate (self facts) if res: culf, facts, codd(res) def gruny (self, e): facts = self 80 ([fr. copracion for f in ent facts]) i=1 print (f' aunging & cg: ') for f in facts: if fact (s) predicate == fact (e), predicate: print(f' 'ttqi'z, qg'') i = 1 auf display (self) print ("Me facts:") for i, g in amenicate (set ([s. copracion for f m self facts])):		elal!	ubian (e))
for in Self, implications ors = i traduate (self facts) if res: culf, facts, codd(res) def gruny (self, e): facts = self 80 ([fr. copracion for f in ent facts]) i=1 print (f' aunging & cg: ') for f in facts: if fact (s) predicate == fact (e), predicate: print(f' 'ttqi'z, qg'') i = 1 auf display (self) print ("Me facts:") for i, g in amenicate (set ([s. copracion for f m self facts])):		suf. facts. ad (fact(e))	
or i waluate (wf. fouts) if no. suf, facts, codd(or w) def growy (suf, e): facts = set \$00 ([fr. commission for f in ent facts]) i=1 print (5' Occurring \$ c 3: ') for f in facts: if fact (3) predicate == fact (e), predicate: print(5' \(\frac{1}{2} \), \(\frac		for in Self, implications	
def group (suf, e): facts = set Pro ([fr. coparation for f in ent fout]) print (f' Guerring & c3: ') for f in facts: print(f' \(\frac{1}{2}\) predicate == fact (e) predicate: print(f' \(\frac{1}{2}\) \) \(\frac{1}{2}\) \(\frac{1}\) \(\frac{1}{2}\)		oes = i. lualvate (ut . lasta)	
dif gury (suf, e): facto = set Po ([fr. copracion for f in ext. facto]) i=1 print (f' Occurring & e3: ') for f in facto: if fact (3) predicate == fact (e), predicate: print (f' \tailing		y su:	
def growy (suf, e): facto = set Pro ([fr. copration for f in set facto]) i=1 print (f' Occarring & e3: ') for f in facto: if fact (3) predicate == fact (e), predicate: print(f' \tail_{1}, \for \for \for \for \for \for \for \for		self, facts, add(ous)	
facts = set Po ([fr. coprasion for f in set facts]) i=1 print (f' Occuping & e 3: ') for f in facts: 'y fact (3) predicate == fact (e), predicate: print(f' \(\t \) \(def gury (suf, e):	
print (5' Querying & c3: ') for f in facts: If fact (3) predicate == fact (e), predicate: print(f' (t2i2, 253')) (1 = 1 def display (suf) print ("Me facts:") for i, 5 in commerce (set (15, expression for 1 m) set, facts))):	100	l_{α}	104 . Jan. 77
for f in facts: if fact (3) producate := fact (e), producate: privat(f '\t\fi\f_1, \formall f\f_3') af display (suf) privat ("All facts:") for i, f in enumerate (set (15, expression for f m set; facts)):		izi	
for f in facts: if fact (3) producate := fact (e), producate: privat(f '\t\fi\f_1, \formall f\f_3') af display (suf) privat ("All facts:") for i, f in enumerate (set (15, expression for f m set; facts)):		print (f' Overging &cg: 1)	
print(f'(t\(\frac{1}{2}\),\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\		for f in facts:	
print(f'(t\(\frac{1}{2}\),\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\		if fact (3) predicate == fact	(c) medicali:
af display (suf) print ("Mu feuts:") for 1, 5 in cumunicate (set (13, expression for 3 m) ext. fouts)):		point(4 '12317, 583')	
for i, 3 in commercia (set (13. expression for 3 m			
for i, 3 in commercia (set (13. expression for 3 m	$\rightarrow \parallel$	alf display (suf)	
for i, 3 in commonts (set (13. common for 3 m			
est feute 300:		되었다면 얼마면 어린 일반에 이번 집에서는 사람이 되고 있다. 그는 사람이 되는 사람이 되었다. 그는 사람이 되었다면 하는 사람이 되었다면 하는 것이 되었다면 하는 것이 없는 것이 없다면 하는 것이 없다.	ion for two
Son Chilling (87)		out Auto 300	
	ولي	ENE (PALESTA SAY)	
	- 200 m		