V SFC V Complite as 5 L V gui © Gome © View V game © Species © Cards © Deck © Piscarlfile © Position © Arboretum © Player V Event © Turn © Rightoscre © get scre © Gareplay			
© Crame © View  V game © Species © Cards © Deck © Piscardfile © Position © Arboretum © Player	1/Src		
© Crame © View  V game © Species © Cards © Deck © Piscardfile © Position © Arboretum © Player	\( (-\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \		
© Crame © View  V game © Species © Cards © Deck © Piscardfile © Position © Arboretum © Player	V COMPILLO JUI) S		
© View V game  © Species © Cards © Deck © Piscardfile © Position © Arboretum © Player			
© View V game  © Species © Cards © Deck © Piscardfile © Position © Arboretum © Player	© Grame		
V game  © Species  © Cards  © Deck  © Piscardfile  © Position  © Arboretum  © Player			
© Piscardfile © Position © Arboretum © Player	V game		
© Piscardfile © Position © Arboretum © Player	(E) Species		
© Piscardfile © Position © Arboretum © Player	© Cards		
© Piscardfile © Position © Arbonetum © Player V Erent © Turn © Rightoscre © getscare © Ganeplay	© Deck		
© Position © Arboretum © Player V Event © Turn © Rightoscre © Gareplay	(2) Discard Dilo		
© Arboretum © Player V Event © Turn © Rightoscre © getsocre © Ganeplay	(C) Position		
© Player  V Event  © Turn  © Righttosocre  © Gareplay	E Arbontum		
V Event © Turn © Rightoscere © get socre © Ganeplay	C Player		
© Turn  © RightToSocre  © GetSocre  © Gancplay	VEront		
© Right To Socre © Get Socre © Gancplay	(E) Turn		
© getsocre  © Ganeplay	C Piglet To Corre		
C Gancplay	C Opp C-cro		
	Constitute of the second of th		
	C) Ciameria		

VSrC	
V compillo, assz	
Vgui	
Viewer	
Game	
y game	
Tenum Species ( A to H	
A + 3 H	
A	
3	
String String int	
Iclass: card (name, species, number)?	
( Construtor: S	
this, name	
this, Species	
this, Species this, number	
7	
Methods:	
get Name 1) )	
apt Spedes() (	
getNamel) (3)  get Species () (1)  get Number () (1)	
9 et Number () ()	
2	

class: deck (Species Num) { Constructor ! for species: species/luns for int 1 to 8 List card this, deck, add (New. Card()); Methods get cards random int from deck. length () >
return Card from deck. pop (int) is Empty () class: discard lile () { Construction: New List (Card) Method: get Cards ()

class.	Dosition	$(\chi, g)$	$) \langle    $	
Cons	tructor: {			
	15 X			
"	15.4			
t	115.9			
}				
Met				
	get X ( )			
	get X ()			
	get (C)			
1				
5				
2		7		
(655-	Arhonotun	n (size)		
Const	rutor (			
	inti	inti		
	inti	X Size	Pos	ition Card
	this, H	ash/yap.put	(Position	(ij), hull);
)				
MADI	rols:			
		Card		
Q	d Card	card, p	osition)	

if Pos Can Place (Position) > Ture/False. Show Current Cards () -> (Postion, Card) Show Avallable Place () - Set of postion) Calculate Score (Specles) int Score reset Map ( ) - ofter first card insert Class: player ( Construtor: ( this, name this arboretum = New Arboretum this discardlile - New Discardlile this, hand cards -> list (card) = Null Methods: { get Name(); get Arboretum (); get Discard Pilet); get Hands (and ();

discard (card)
draw (location);
play (card, position);
Y Event
class: Turn (player, Deck)
player, drav()
player. dran ()
player. play ()
player des carde)
class: Right To Scare ( players)
foreach player: players
- which is higher
hashMap return player A: 151, 52, 53-7
Player B: (54, 55, 56)
Player C:

Tologo, Gat Scarce / playbox list (paging)
class: get Score (Player, List (species))
Socre = 0
for each specie: species (
Socre += player, get Amoretum, calculat scroe (sped)
3
return govre
Class: Game Play ( ) S
New Deck (G)
New player (A)
Yew player (B)
while ( Deck istimpty) }
New Turn (PlayerA, Deck);
/Yew Turn (Player B. Deck);
i+ (Deck. is Empty)
map = Right To Score ( List=( Player A, Player B));
foreach i ; map (
get Sore (i[o], i[i]);
2
-> compare to vinner?