# Script Notes

import: \* from family

create lists: male migration, new family counter (counts number of individuals in new family after a family reaches > 45 members and splits up; stops at 23 members, or half of 46), new male family counter (same thing, but creates smaller all-male groups), old family ids (stored for archival purposes), new families.and new male families dictionary (to both store unique IDs and count at once)

class Monkey(agent):

initialize attributes (unique ID, model reference, gender, age, age\_category, family, last birth interval, mother ID)

define step() function:

Loop through each monkey individual at each step

Check age category

Age (+= 1/73 years on each 5-day step)

Update reproductive female and recent-infant-death lists

Birth from the mother’s perspective; if 49 < self.model.step\_in\_year < 55 and self.gender == 1 (female) and random 8-9 < self.age <= 25 and self.last\_birth\_interval > 3:

set self.family.family\_size += 1

self.birth()

set self.last\_birth\_interval = 0

Death() (chance according to age as below):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Age | 0-1 | 1-10 | 10-30, Male | 10-30, Female | 10-30, Overall | 30+ |
| Yearly Mortality | 20% | 5% | 13% | 5.5% | 9% | 60% |

The table above was adapted from the original version of the pseudocode (which did not differentiate between male and female mortality rates). The above mortality rates should yield the following age/sex structure:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Group | Baby | Juvenile | Young | Sub-adult | Adult | Senior |
| Age thresholds | <1 year | 1 to <3 years | 3 to <7 | 7 to <10 | 10 to <25 | 25 or above |
| % | 11% | 16% | 15% | 20% | 34% | 4% |
| F:M ratio at each age group | 1:1 | 1:1 | 1:1 | 1:1 | 2.9:1 | All females |

Check if all-male subgroup needs to break off of the main group (enough males accumulate)

Check if a family group is too large (from an individual’s perspective) and needs to split; if it does (family\_size > 46):  
 Create new family

If a family is currently splitting:

Continue splitting until self.family.family\_size >= 24, at which time family\_split\_flag changes and it stops splitting

Update new\_family\_list, self.family\_id

Migrate to new family if current family if splitting

define age\_category() function:

Age categories are as follows, with the higher bound of the age being inclusive: age < 1, 1-3, 3-7, 7-10, 10-25, and 25+ (this differs from the death probability table above). Monkeys are considered of breeding age above 8-9; male monkeys can join an all-male subgroup starting at age 10, and monkeys are considered of old age above 25.

The demographic structure list is updated accordingly.

define check\_recent\_death\_infant() function:

Mothers who have recently lost an infant monkey will be in the recent\_death\_infant list (from the infant’s perspective, if they die, their mother’s unique ID becomes added to the list). This checks who is on the list, gives them a renewed self.last\_birth\_interval of 2-2.5 years so that they may reproduce again in 6 months-1 year, then removes them from the list.

define birth() function:

creates a monkey infant with the following attributes: unique ID, model reference, gender (randomized), age, age\_category, family ID, last\_birth\_interval, mother ID

self.model.schedule.add(new\_monkey)  
self.model.number\_of\_monkeys += 1  
self.model.monkey\_id\_count += 1  
self.model.monkey\_birth\_count += 1  
demographic\_structure\_list[0] += 1

define death() function:

decrease family\_size, remove family from schedule and grid if this is the last monkey in their family, decrease # of monkeys in reserve, increase death count, remove monkey from any attribute list it belongs to (e.g. reproductive\_female\_list), remove monkey agent from schedule

define create\_new\_family() function:

from model import global\_family\_id\_list  
assign a new family id, append it to the global\_family\_id\_list, adopt the new family’s current\_position from the current family it is splitting from, set the split\_flag, set the new family type (all-male or traditional, depending on the split conditions), create the new family agent and place on the grid and schedule, and increase the number of families. This function also returns the new\_family itself.

define migrate\_to\_new\_family() function:

self.family.family\_size -= 1  
if self.unique\_id in self.family.list\_of\_family\_members:  
 self.family.list\_of\_family\_members.remove(self.unique\_id)  
self.family = new\_family  
self.family.list\_of\_family\_members.append(self.unique\_id)  
self.family.family\_size += 1