STATISTICAL RETHINKING 2023
WEEK 8 SOLUTIONS

1. The dyad model from the book and lecture can be used almost unmodified, except the probability of the like nominations should be binomial, because it has a known maximum of 3. Here is the modified model:

```
library(rethinking)
# if you have the file:
# dat <- read.csv("week08_Monks.csv")
# otherwise
data(Monks)
dat <- Monks

# dyad model
f_dyad <- alist(
    likeAB ~ binomial( 3 , pAB ),
    likeBA ~ binomial( 3 , pBA ),
    logit(pAB) <- a + T[D,1] ,
    logit(pBA) <- a + T[D,2] ,
    a ~ normal(0,1),

    ## dyad effects - non-centered
    transpars> matrix[N_dyads,2]:T <-
            compose_noncentered( rep_vector(sigma_T,2) , L_Rho_T , Z ),
    matrix[2,N_dyads]:Z ~ normal( 0 , 1 ),
    cholesky_factor_corr[2]:L_Rho_T ~ lkj_corr_cholesky( 2 ),
    sigma_T ~ exponential(1),

    ## compute correlation matrix for dyads
    gq> matrix[2,2]:Rho_T <<- Chol_to_Corr( L_Rho_T )
)

n_dyads <- nrow(dat)
monk_dat <- list(
    N_dyads = nrow(dat),
    D = dat$dyad_id,
    likeAB = dat$like_AB,
    likeBA = dat$like_BA )

m1 <- ulam( f_dyad , data=monk_dat , chains=4 , cores=4 , iter=2000 )
precis(m1,depth=3,pars=c("Rho_T","sigma_T"))
```

```
            mean   sd 5.5% 94.5% n_eff Rhat4
Rho_T[1,1] 1.00 0.00 1.00  1.00   NaN   NaN
Rho_T[1,2] 0.62 0.11 0.43  0.78  1096     1
Rho_T[2,1] 0.62 0.11 0.43  0.78  1096     1
Rho_T[2,2] 1.00 0.00 1.00  1.00   NaN   NaN
sigma_T    3.41 0.42 2.77  4.13  1569     1
```

So the posterior correlation within dyads is 0.43 – 0.78. Is that a lot of reciprocity? Hard to say. We haven't done anything to deal with properties of the monks themselves and generalized liking. Also these data were collected by asking for top 3 likes. So it could be that there is more reciprocity in the total network, outside top 3 likes.

2. This model just needs dislikes sections that mimic the code for the like sections.

```
# dyad model with dislike
f_dyad2 <- alist(
    likeAB ~ binomial( 3 , pAB ),
    likeBA ~ binomial( 3 , pBA ),
    logit(pAB) <- a[1] + T[D,1] ,
    logit(pBA) <- a[1] + T[D,2] ,

    dislikeAB ~ binomial( 3 , qAB ),
    dislikeBA ~ binomial( 3 , qBA ),
    logit(qAB) <- a[2] + T2[D,1] ,
    logit(qBA) <- a[2] + T2[D,2] ,
    vector[2]:a ~ normal(0,1),

    ## like dyad effects - non-centered
    transpars> matrix[N_dyads,2]:T <-
            compose_noncentered( rep_vector(sigma_T,2) , L_Rho_T , Z ),
    matrix[2,N_dyads]:Z ~ normal( 0 , 1 ),
    cholesky_factor_corr[2]:L_Rho_T ~ lkj_corr_cholesky( 2 ),
    sigma_T ~ exponential(1),

    ## dislike dyad effects - non-centered
    transpars> matrix[N_dyads,2]:T2 <-
            compose_noncentered( rep_vector(sigma_T2,2) , L_Rho_T2 , Z2 ),
    matrix[2,N_dyads]:Z2 ~ normal( 0 , 1 ),
    cholesky_factor_corr[2]:L_Rho_T2 ~ lkj_corr_cholesky( 2 ),
    sigma_T2 ~ exponential(1),

    ## compute correlation matrix for dyads
    gq> matrix[2,2]:Rho_T <<- Chol_to_Corr( L_Rho_T ),
    gq> matrix[2,2]:Rho_T2 <<- Chol_to_Corr( L_Rho_T2 )
)

n_dyads <- nrow(dat)
monk_dat <- list(
    N_dyads = nrow(dat),
    D = dat$dyad_id,
    likeAB = dat$like_AB,
    likeBA = dat$like_BA,
    dislikeAB = dat$dislike_AB,
    dislikeBA = dat$dislike_BA )

m2 <- ulam( f_dyad2 , data=monk_dat , chains=4 , cores=4 , iter=2000 )
precis( m2 , depth=3 , pars=c("Rho_T","Rho_T2") )
```

```
              mean   sd 5.5% 94.5% n_eff Rhat4
Rho_T[1,1]    1.00 0.00 1.00  1.00   NaN   NaN
Rho_T[1,2]    0.63 0.11 0.44  0.79  1050  1.01
Rho_T[2,1]    0.63 0.11 0.44  0.79  1050  1.01
Rho_T[2,2]    1.00 0.00 1.00  1.00   NaN   NaN
Rho_T2[1,1]   1.00 0.00 1.00  1.00   NaN   NaN
Rho_T2[1,2]   0.45 0.14 0.22  0.66   872  1.00
Rho_T2[2,1]   0.45 0.14 0.22  0.66   872  1.00
Rho_T2[2,2]   1.00 0.00 1.00  1.00   NaN   NaN
```

The dislike correlation is a bit smaller. Let's compute the contrast however:

```
post <- extract.samples(m2)
round(quantile( post$Rho_T[,1,2] - post$Rho_T2[,1,2] ),2)
```

```
   0%   25%   50%   75%  100%
-0.43  0.07  0.18  0.29  0.84
```

So more than 75% of the posterior mass is positive, indicating that reciprocity for liking is higher than for disliking. But there's a lot of uncertainty in both estimates, so still lots of uncertainty in this contrast.

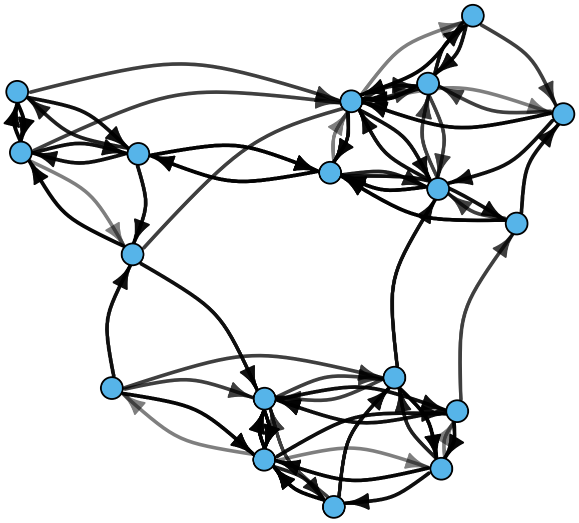
For fun, let's visualize these networks.

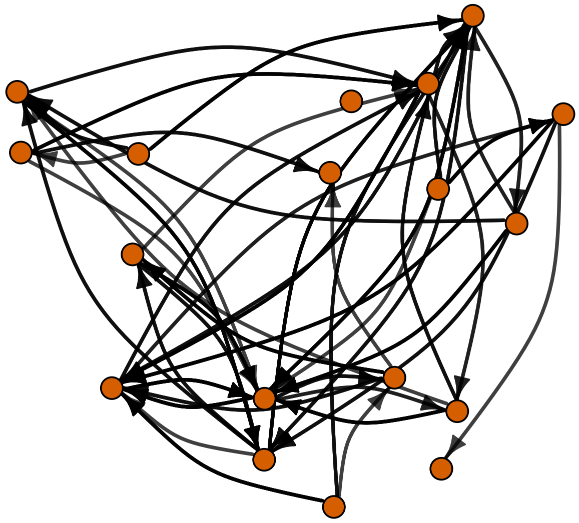
```
T_est <- apply(post$T,2:3,mean)
T2_est <- apply(post$T2,2:3,mean)

N <- 18
n_dyads <- choose(18,2)
dyads <- t(combn(18,2))
# convert to adjacency matrix
y_est <- matrix(NA,18,18)
y2_est <- matrix(NA,18,18)
for ( i in 1:n_dyads ) {
    y_est[ dyads[i,1] , dyads[i,2] ] <- T_est[i,1]
    y_est[ dyads[i,2] , dyads[i,1] ] <- T_est[i,2]
    y2_est[ dyads[i,1] , dyads[i,2] ] <- T2_est[i,1]
    y2_est[ dyads[i,2] , dyads[i,1] ] <- T2_est[i,2]
}#i

library(igraph)
# like
sng <- graph_from_adjacency_matrix(y_est)
lx <- layout_nicely(sng)
plot(sng , layout=lx , vertex.size=8 , edge.arrow.size=1 ,
edge.width=4 , edge.curved=0.35 , vertex.color=2 , edge.color=grau(0.2) ,
asp=0.9 , margin = -0.05 , vertex.label=NA )

# dislike
sng2 <- graph_from_adjacency_matrix(y2_est)
plot(sng2 , layout=lx , vertex.size=8 , edge.arrow.size=1 ,
edge.width=4 , edge.curved=0.5 , vertex.color=6 , edge.color=grau(0.2) ,
asp=0.9 , margin = -0.05 , vertex.label=NA )
```

The networks are on the next page. The blue one is the liking network, arranged so it is clustered by reciprocal liking. The orange one is the disliking network, using the same layout as the liking network, so you can more easily compare. It's clear that the disliking tends to point at other clusters in the network. And a few individuals tend to be nominated a lot.

3. To include generalized receiving of likes and dislikes, we can use the same structure as in the book example, but now both dimensions of the matrix are receiving: the first is likes and the second is dislikes. Having both general receiving effects in the same distribution allows us to estimate the correlation across networks.

```
# general receiving model
f_dyad3 <- alist(
    likeAB ~ binomial( 3 , pAB ),
    likeBA ~ binomial( 3 , pBA ),
    logit(pAB) <- a[1] + T[D,1] + R[B,1],
    logit(pBA) <- a[1] + T[D,2] + R[A,1],

    dislikeAB ~ binomial( 3 , qAB ),
    dislikeBA ~ binomial( 3 , qBA ),
    logit(qAB) <- a[2] + T2[D,1] + R[B,2],
    logit(qBA) <- a[2] + T2[D,2] + R[A,2],
    vector[2]:a ~ normal(0,1),

    ## like dyad effects - non-centered
    transpars> matrix[N_dyads,2]:T <-
            compose_noncentered( rep_vector(sigma_T,2) , L_Rho_T , Z ),
    matrix[2,N_dyads]:Z ~ normal( 0 , 1 ),
    cholesky_factor_corr[2]:L_Rho_T ~ lkj_corr_cholesky( 2 ),
    sigma_T ~ exponential(1),
```





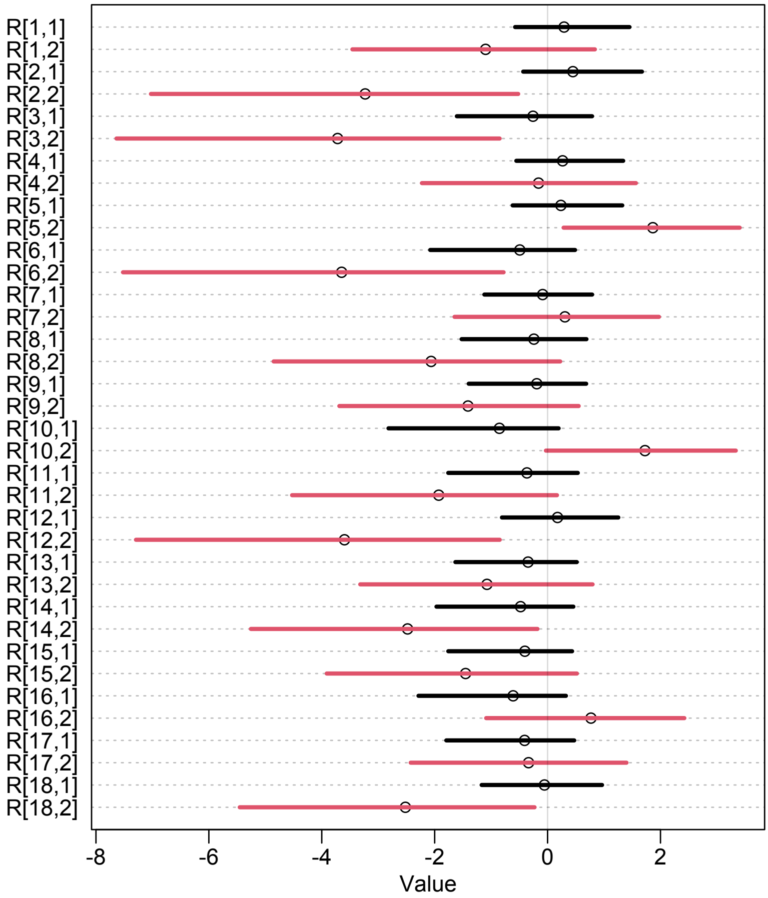
```
    ## dislike dyad effects - non-centered
    transpars> matrix[N_dyads,2]:T2 <-
            compose_noncentered( rep_vector(sigma_T2,2) , L_Rho_T2 , Z2 ),
    matrix[2,N_dyads]:Z2 ~ normal( 0 , 1 ),
    cholesky_factor_corr[2]:L_Rho_T2 ~ lkj_corr_cholesky( 2 ),
    sigma_T2 ~ exponential(1),

    ## R matrix of receiving effects
    transpars> matrix[18,2]:R <-
            compose_noncentered( sigma_R , L_Rho_R , ZR ),
    matrix[2,18]:ZR ~ normal( 0 , 1 ),
    cholesky_factor_corr[2]:L_Rho_R ~ lkj_corr_cholesky( 2 ),
    vector[2]:sigma_R ~ exponential(1),

    ## compute correlation matrix for dyads
    gq> matrix[2,2]:Rho_T <<- Chol_to_Corr( L_Rho_T ),
    gq> matrix[2,2]:Rho_T2 <<- Chol_to_Corr( L_Rho_T2 ),
    gq> matrix[2,2]:Rho_R <<- Chol_to_Corr( L_Rho_R )
)

n_dyads <- nrow(dat)
monk_dat <- list(
    N_dyads = nrow(dat),
    D = dat$dyad_id,
    likeAB = dat$like_AB,
    likeBA = dat$like_BA,
    dislikeAB = dat$dislike_AB,
    dislikeBA = dat$dislike_BA,
    A = dat$A,
    B = dat$B )
m3 <- ulam( f_dyad3 , data=monk_dat , chains=4 , cores=4 , iter=2000 )
precis( m3 , depth=3 , pars=c("Rho_T","Rho_T2","sigma_R","Rho_R") )
```

```
             mean   sd  5.5% 94.5% n_eff Rhat4
Rho_T[1,1]   1.00 0.00  1.00  1.00   NaN   NaN
Rho_T[1,2]   0.65 0.11  0.47  0.81  1236  1.00
Rho_T[2,1]   0.65 0.11  0.47  0.81  1236  1.00
Rho_T[2,2]   1.00 0.00  1.00  1.00   NaN   NaN
Rho_T2[1,1] 1.00 0.00  1.00  1.00   NaN   NaN
Rho_T2[1,2] 0.53 0.16  0.26  0.77  1134  1.00
Rho_T2[2,1] 0.53 0.16  0.26  0.77  1134  1.00
Rho_T2[2,2] 1.00 0.00  1.00  1.00   NaN   NaN
sigma_R[1]   0.71 0.51  0.07  1.63   682  1.00
sigma_R[2]   2.51 0.92  1.36  4.24  1007  1.00
Rho_R[1,1]   1.00 0.00  1.00  1.00   NaN   NaN
```

```
Rho_R[1,2]  0.02 0.40 -0.63  0.66   411  1.01
Rho_R[2,1]  0.02 0.40 -0.63  0.66   411  1.01
Rho_R[2,2]  1.00 0.00  1.00  1.00   NaN   NaN
```

First note that the dyad reciprocity for disliking has increased a little, as a result of modeling the generalized disliking. And the standard deviation that corresponds to general disliking, `sigma_R[2]`, is pretty large. This is consistent with some monks getting a lot dislike nominations than others. The likes network doesn't seem to show the same phenomenon. Things are more balanced there. Finally, there doesn't seem to be any evidence of a correlation in these general effects across the networks: `Rho_R[1,2]` is nothing much.

We can see the different patterns in general likes/dislikes by plotting the posterior distributions for the `R` matrix of effects. It looks like this:



Monks are listed from top to bottom (the first index), and the red intervals are the dislike elements. A few monks, 5 and 10 and 16, have large values. The rest are pretty much ignored. The black intervals, the liking estimates, are all pretty close to zero.

4-OPTIONAL CHALLENGE. There are a few ways to do this. The approach I will take is the use a block model in which there is a matrix of offsets for an individual from one faction to nominate a member of any other (or the same) faction. So we need two 4-by-4 matrices, one for likes and one for dislikes. The diagonal of each matrix will be nominations within each faction. The off-diagonal elements are nomination between factions. Here is the model code:

```
# block model
f_block <- alist(
    likeAB ~ binomial( 3 , pAB ),
    likeBA ~ binomial( 3 , pBA ),
    logit(pAB) <- a[1] + T[D,1] + bl[F[A],F[B]],
    logit(pBA) <- a[1] + T[D,2] + bl[F[B],F[A]],

    dislikeAB ~ binomial( 3 , qAB ),
    dislikeBA ~ binomial( 3 , qBA ),
    logit(qAB) <- a[2] + T2[D,1] + bd[F[A],F[B]],
    logit(qBA) <- a[2] + T2[D,2] + bd[F[B],F[A]],

    vector[2]:a ~ normal(0,1),
    matrix[4,4]:Zl ~ normal(0,1),
    matrix[4,4]:Zd ~ normal(0,1),
    c(tau_l,tau_d) ~ half_normal(0,1),
    transpars> matrix[4,4]:bl <<- Zl*tau_l,
    transpars> matrix[4,4]:bd <<- Zd*tau_d,

    ## like dyad effects - non-centered
    transpars> matrix[N_dyads,2]:T <-
            compose_noncentered( rep_vector(sigma_T,2) , L_Rho_T , Z ),
    matrix[2,N_dyads]:Z ~ normal( 0 , 1 ),
    cholesky_factor_corr[2]:L_Rho_T ~ lkj_corr_cholesky( 2 ),
    sigma_T ~ exponential(1),

    ## dislike dyad effects - non-centered
    transpars> matrix[N_dyads,2]:T2 <-
            compose_noncentered( rep_vector(sigma_T2,2) , L_Rho_T2 , Z2 ),
    matrix[2,N_dyads]:Z2 ~ normal( 0 , 1 ),
    cholesky_factor_corr[2]:L_Rho_T2 ~ lkj_corr_cholesky( 2 ),
    sigma_T2 ~ exponential(1),

    ## compute correlation matrix for dyads
    gq> matrix[2,2]:Rho_T <<- Chol_to_Corr( L_Rho_T ),
    gq> matrix[2,2]:Rho_T2 <<- Chol_to_Corr( L_Rho_T2 )
)
```

The `F` variable is the faction list. The `bl[F[A],F[B]]` code in the linear model pulls out the element of the liking matrix correspond to an individual from A’s faction liking someone from B's faction on average. Similarly bd[F[A],F[B]] extracts the offset for someone from A's faction disliking someone from B's faction. The elements of these matrices are partially pooled within each matrix, and I use a non-centered parameterization. Let's run the chains:

```
# add factions
factions <- c( 1 ,1, 3, 2, 2, 2, 1, 4, 2, 4, 2, 1, 4, 1, 1, 1, 3, 3 )
monk_dat <- list(
    N_dyads = nrow(dat),
    D = dat$dyad_id,
    likeAB = dat$like_AB,
    likeBA = dat$like_BA,
    dislikeAB = dat$dislike_AB,
    dislikeBA = dat$dislike_BA,
    A = dat$A,
    B = dat$B,
    F = factions )
m4 <- ulam( f_block , data=monk_dat , chains=4 , cores=4 , iter=2000 )
precis( m4 , depth=3 , pars=c("a","Rho_T","sigma_T","Rho_T2",
"sigma_T2","tau_l","tau_d") )
```

```
              mean   sd  5.5% 94.5% n_eff Rhat4
a[1]         -3.14 0.65 -4.15 -2.07  2141     1
a[2]         -4.34 0.49 -5.14 -3.58  3271     1
Rho_T[1,1]    1.00 0.00  1.00  1.00   NaN   NaN
Rho_T[1,2]    0.36 0.16  0.07  0.61  1670     1
Rho_T[2,1]    0.36 0.16  0.07  0.61  1670     1
Rho_T[2,2]    1.00 0.00  1.00  1.00   NaN   NaN
sigma_T       2.59 0.37  2.04  3.22  2093     1
Rho_T2[1,1]   1.00 0.00  1.00  1.00   NaN   NaN
Rho_T2[1,2]   0.46 0.14  0.22  0.67  1455     1
Rho_T2[2,1]   0.46 0.14  0.22  0.67  1455     1
Rho_T2[2,2]   1.00 0.00  1.00  1.00   NaN   NaN
sigma_T2      3.36 0.43  2.70  4.09  2309     1
tau_l         2.49 0.52  1.73  3.37  2592     1
tau_d         0.70 0.48  0.07  1.58  1019     1
```

I haven't displayed the faction offsets here. We'll look at those next. But look at the tau variance components for the blocks. The large standard deviation for the liking network tau_l suggests that the elements of the liking matrix vary a lot. The low tau_d suggests the opposite for disliking. Note also that the correlation within dyads for the liking network, Rho_T[1,2, has dropped a lot. That suggests the factions do explain a lot of the reciprocity within dyads. That is to say that the liking network has reciprocity with the faction.

Let's look at the bl and bd faction offset matrices.

```
post <- extract.samples(m4)
( BL <- round(apply(post$bl,2:3,mean),2) )
( BD <- round(apply(post$bd,2:3,mean),2) )
```

```
> ( BL <- round(apply(post$bl,2:3,mean),2) )
      [,1]  [,2]  [,3]  [,4]
[1,]  1.45 -4.48 -2.35 -3.87
[2,] -2.58  1.71 -3.47 -0.65
[3,] -1.28 -3.49  3.54 -1.22
[4,] -2.78  1.24 -0.88 -0.59
> ( BD <- round(apply(post$bd,2:3,mean),2) )
      [,1]  [,2]  [,3]  [,4]
[1,] -0.49 -0.16 -0.67  0.02
[2,] -0.47 -0.09 -0.34  0.62
[3,]  0.15 -0.04  0.11 -0.44
[4,] -0.45 -0.02 -0.47 -0.32
```

The top matrix is the liking network. Notice that the diagonal values are larger than the off-diagonal values, at least for factions 1, 2 and 3. Faction 4 is the weird one. It has a larger offset for nominating members of faction 2 than members of its own faction.

The bottom matrix is the disliking network. Here the larger values are off-diagonal. The monks tend to dislike monks in other factions. But there is no strong pattern here. Note however that faction 2 does weakly tend to dislike members of faction 2, in reverse of what we saw in the liking network. But it is very weak here. The story about disliking is more a story of individuals and not factions.

Okay, finally let's plot the networks. We need to include both the faction offsets and the dyad effect in computing network ties.

```
N <- 18
n_dyads <- choose(18,2)
dyads <- t(combn(18,2))
# convert to adjacency matrix
y_est <- matrix(NA,18,18)
F <- factions
for ( i in 1:n_dyads ) {
    A <- dyads[i,1]
    B <- dyads[i,2]
    y_est[ A , B ] <- mean( inv_logit(
        with( post , a[,1]+T[,i,1]+bl[,F[A],F[B]] ) ) )
    y_est[ B , A ] <- mean( inv_logit(
        with( post , a[,1]+T[,i,2]+bl[,F[B],F[A]] ) ) )
}#i
# like
sng <- graph_from_adjacency_matrix(3*y_est/max(y_est,na.rm=TRUE))
```

```
lx <- layout_nicely(sng)
plot(sng , layout=lx , vertex.size=8 , edge.arrow.size=0.7 ,
edge.width=2 , edge.curved=0.35 , vertex.color=factions ,
edge.color=grau() , asp=0.9 , margin = -0.05 , vertex.label=NA )
# dislike
sng2 <- graph_from_adjacency_matrix(y2_est)
plot(sng2 , layout=lx , vertex.size=8 , edge.arrow.size=0.7 ,
edge.width=2 , edge.curved=0.35 , vertex.color=factions ,
edge.color=grau() , asp=0.9 , margin = -0.05 , vertex.label=NA )
```

I show the networks on the next page. The liking network is on top. The disliking network is on the bottom. In both, I've colored the nodes by faction. And again I've placed the nodes in both networks according the liking network ties. You can see that the liking network is really focused on the factions. The disliking network is not. Although there does appear to be one individual in each of the orange, blue, and yellow factions who is widely disliked by members of other factions.