1. Because N\_a\_new is smaller than the original population of the wild-type because some will mutate from a (wildtype) to A (new mutant), whereas it has to be added to N\_A because this value gets bigger.
2. Individuals reproduce first and then they mutate
3. High mutation rate in the model 🡪0.01🡪1% of mutation
4. C returns the new values for N\_a and N\_A after one generation  
   c: puts it in to the object that is then returned 🡪 R can not deal with two separate values  
   max: makes sure that it never gets negative (not really needed now)
5. Pop\_vector is just the ouput vector that we create to store our results and pop\_new is to adjust for the new population that we get one generation later, which is then stored in pop\_vector
6. The condition is that the loop is broken if the population gets too big (1.5 the initial population) which is good because then the simulation can not go on for too long, but it also doesn’t allow us to look far ahead in the future, might get tables of different lengths (issues with coding later on, hard to analyse together)  
   it also stops when the population goes extinct which is good because then we have no individuals left
7. It is to only show a part of the vector, last bit, helpful because: see if something happens at the end, show unexpected results too
8. But black-line on top of other lines in the report

We now allow the values for r (decay rate) and s (selection coefficients) to vary and for each of those values we take 100 replicates, from this we analyse the minimum population size and at which generation this occurs

1. Because we run it for a longer time, so for each set of parameters we say it can vary and then we run 100 separate replicates  
   r\_values and s\_values both have different numbers (in the vectors) and for each one of them, 100 replicates are done
2. Because if the value changes we could get confused and use the wrong number, this way
3. which(total\_size==min\_size)[1]

before we determined the minimum population size, with the which function we want to see at which generation the population has his minimum population size  
if no\_min=0 than the population is extinct

second deleterious mutation: change s and mutation rate  
which of them goes to extinction   
r: decay rate probably fixed   
s1 (beneficial):positive  
s2 (deleterious): negative?  
mutation-rate would probably have to be the same for both   
1 plot where we change mutation rate not s, 1 plot where we change s but not mutation rate  
🡪 would still be one population

1 beneficial, 1 deleterious   
initial frequencies of mutations, analyse time (generations) to extinction  
different selection coefficients, different initial frequencies for both separate   
measure generations to extinction, how many get extinct (counter at the end of the loop)  
population size fixed  
can s be minus?

2 mutations: 1 deleterious one with selection coefficients (maybe 0.1-0.4) and a second one which is less deleterious, with s (between 0.5-0.9)  
these mutations mutate to beneficial ones, which one wins?  
populations-size fixed   
at the beginning: their frequencies are 50/50

What do you want to know?  
is the trade-off between having a really bad mutation in the population and mutating to a very beneficial one better or is it better to be slightly deleterious in the beginning and mutate to a slightly beneficial one

How do 2 deleterious mutations which can mutate to beneficial ones influence the rescue probability of a population and how does their frequency in the population change?

Project:

Variables:

* N\_wm (weak mutation), N\_sm (strong mutation)
* s\_coeff ?

Constraints on the variables

* s\_coeff\_weak (-1🡪+1)
* s\_coeff\_strong (-1🡪+1)

Interactions between variables

Time discrete or continuous?

* Discrete generations

Choose a time scale (generations, minutes, etc.)

* Number of generations

Define the parameters

* Decay\_rate\_wm, decay\_rate\_sm
* m\_rate\_wm, m\_rate\_sm
* t\_max
* init\_wm=50, init\_sm=50
* s\_wm, s\_sm

Constraints on the paramteters (eg. Between 0 and 1)

* decay\_rate\_wd=0.1🡪taken from the base-model
* decay\_Rate\_sd=?
* mut\_rate\_wd, mut\_rate\_sd (0-1)
* t\_max=1000
* no\_replicates=100