No quiz next week (dead week). Ideally the homework should be turned in before dead week, i.e. midnight Saturday, but we’ll do it after class. Tuesday is the last day of class, there will be classwork.

**Picking a random integer between n, m**

The rand() doesn’t work for the online compiler, instead of r=rand() write call random_number(r).

\[
\begin{align*}
& r \in [0, 1), \\
& (k+1) \times r \in [0, k+1) = [0, 1) \cup [1, 2) \cup \ldots \cup [k, k+1), \\
& \text{floor}((k+1) \times r) \in \{0, 1, 2, \ldots, k\}, \\
& \text{floor}(m+(k+1) \times r) \in \{m, m+1, \ldots, m+k\}.
\end{align*}
\]

Suppose \( m \leq n \). Let \( k = n - m \), then \( m + k = n \). Hence \( \text{floor}(m+(n-m+1) \times r) \in \{m, m+1, \ldots, n\} \). The following function randint randomly picks an integer from \( \{m, m+1, \ldots, n\} \).

```latex
integer function randint(m, n) 
integer::m, n; real::r 
call random_number(r) 
randint = floor(m+(n-m+1)*r) 
endfunction
```
**Running Averages, Probabilities.**

As numbers come in, one at a time, we want to compute a running average by updating the average as values come in without having to keep a record of the previous values. Suppose the values are $a_1, a_2, a_3, \ldots, a_n$ The $n$th average is

$$A_n = \frac{a_1 + a_2 + a_3 + \ldots + a_{n-1} + a_n}{n}$$

The previous average is $A_{n-1} = \frac{a_1 + a_2 + a_3 + \ldots + a_{n-1}}{n - 1}$.

Multiply by $n - 1$: $A_{n-1}(n - 1) = a_1 + a_2 + a_3 + \ldots + a_{n-1}$.

Add $a_n$: $A_{n-1}(n - 1) + a_n = a_1 + a_2 + a_3 + \ldots + a_{n-1} + a_n$.

Divide by $n$: $\frac{A_{n-1}(n - 1) + a_n}{n} = \frac{a_1 + a_2 + a_3 + \ldots + a_{n-1} + a_n}{n}$.

The last number is the new average $A_n$. Hence the updated average is $A_n = (A_{n-1}(n - 1) + a_n)/n$. In Fortan,

```fortran
average = (average * (n-1) + current_value) / n.
```

Start running averages start at 0.
Event variables: let $p=1$ if the event succeeds and $p=0$ if not. \([0, 1, 1, 0, 0]\) means the event occurred 2 times out of 5 and hence has probability \(2/5\). This probability is also the average \((0+1+1+0+0)/5\) of the event values since the sum \(0+1+1+0+0\) equals the number of event occurrences. Hence the probability $p_p$ of the event is the average of the values of its event variable $p$. The formula for updating this running average is $p_p=(p_p \times (n-1)+p)/n$. To get a good estimate of the probability, use a long-term average of about a million runs, $n=10^6$.

Rolling a die gives a random number between 1 and 6. This is exactly what `randint(1, 6)` does. Consider the event that the die comes up 1 or 2. The probability of this event is $2/6 = .333...$. For more complicated problems, there is no known formula and you have to compute the probability by running a program for the event. Just one roll won’t calculate a probability or an average. To get a good long-term average, roll the die a million times.
Classwork 28.1(3) die_roll.f95 Calculate the probability of the event that a randomly rolled die comes up 1 or 2. Let p be the event variable. Thus p=1 if 1 or 2 comes up, p=0 if not. The probability pp of getting 1 or 2 in a roll of a die will be the long-term average (a million rolls will do) of p.
program roll_die

integer:: randint, die ! die=# that comes up on die
real:: p, pp=0 ! p=event of 1, 2, pp=running probability
integer, parameter:: inf=10**6

do n=1, inf ! do a million rolls to get a long-term average.
    die = randint(1, 6)
    if (die==1 .or. die==2) then
        p = _____
    else
        p = _____
    endif
    pp = ___________________ ! running average
enddo

print*, pp

end program ! ans. .333...

integer function randint(m, n)
integer:: m, n; real:: r

call random_number(r)
randint = floor(m+(n-m+1)*r)
endfunction
**Monty Hall Problem**

**Classwork 28.2(5) montyhall.f95**

There are three closed doors: doors 1, 2, 3. The car is randomly placed behind one of these three doors, hence let \( \text{cardoor} = \text{randint}(1, 3) \). The other two doors have goats.

You choose one of the doors, call it \( \text{firstchoice} \). Monty Hall, the game show host, opens one of the other two doors to reveal a goat.

You now have two choices:
- stay with your first choice or
- switch to the other closed door.

You win if the door you pick has the car.

Which is the better strategy: staying with your first choice or switching? Let \( \text{pstay} = 1 \) if staying with your first choice wins, \( = 0 \) if not. Let \( \text{pswitch} = 1 \) if switching wins, \( = 0 \) if not. To calculate the probabilities of winning by staying with your first choice or by switching, play the game a million times and record the running average \( \text{ppst ay} \) of \( \text{pstay} \) and the average \( \text{ppswitch} \) of \( \text{pswitch} \).
program monty_hall
integer::randint, cardoor, firstchoice=1
integer::pstay, pswitch  ! event variables
real::ppstay=0, ppswitch=0  ! probabilities
integer, parameter::inf=10**6

do n=1, inf  ! play game a million times
  cardoor= __________  ! randomly place car
  if (cardoor==firstchoice) then
    pstay= _____ ; pswitch= _____
  else
    pstay= _____ ; pswitch= _____
  endif
  ppstay = _____________________________
  ppswitch=(ppswitch*(n-1)+pswitch)/n
enddo

print*, 'Prob. staying wins:', ppstay
print*, 'Prob. switching wins:', ppswitch

end program

integer function randint(m,n)
call random_number(r)
randint = floor(m+(n-m+1)*r)
end function
A drunk starts at the origin $w=[0,0]$. Each step is randomly 1 unit to the right, 1 unit left, 1 unit up or 1 unit down. On average, how far is he from the origin after 100 steps? The complicated formula for the average distance after $N$ steps involves gamma functions; it is a little less than $\sqrt{N}$. For 100 it is about 9.
program random_walk
real:: distance, aver_dist = 0
integer:: w(2), direction(4, 2), randint
integer, parameter:: inf = 10**4

direction(1,:) = (/1, 0/)  ! right one step
direction(2,:) = (-1, 0/)  ! left one step

direction(3,:) = (/_______/)  ! up one step
direction(4,:) = (/_______/)  ! down one step

do n = 1, inf
    w = (/_______/
    do i = 1, ______
        w = ______________________
    enddo
    distance = sqrt(real(sum(w**2)))
    aver_dist = (aver_dist*(n-1)+distance)/n
endo

3 format('Average distance after', a, f5.2)
print 3, ' 100 steps:', aver_dist
endprogram

integer function randint(m, n)
call random_number(r)
randint = floor(m+(n-m+1)*r)
endfunction
Write a subroutine `shuffle(v, n)` which shuffles the n elements of the vector v. This means we must randomly permute the elements. Suppose the elements are integers. If the vector is a deck of cards `cards=[1,2,3,...,52]`, then the shuffled deck might look like `cards=[28,4,49,33,8,...,6]`. 
recursive subroutine shuffle(cards,n)
integer::cards(n),randint
!base case, n=1, only one possible arrangement
if(n==1) return
!recursion case
j = ___________ !randomly choose an index between 1 and n
!put the card at the random position j into position 1
call swap(cards(_____),cards(_____))
!recursively call shuffle to
!shuffle the remaining n-1 cards in positions 2 to n.
call shuffle(cards(_____),_____
endsubroutine
integer function randint(m,n)
call random_number(r)
randint = floor(m+ (n-m+1)*r)
endfunction
subroutine swap(n,m)
integer::n,m,k
k=n; n=m; m=k
endsubroutine
program shuffle_card_deck
integer::cards(52)
do i=1,52;cards(i)=i;enddo
print*,'First 13 cards of 7 shuffles.'
do i=1,7
    call shuffle(cards,52)
    print '(13(i3),a)',(cards(j),j=1,13)," ..."
endo;
endprogram

**Homework 28.2(4) 3Drandomwalk.f95** A drunk (angel?) starts at the origin $w=[0,0,0]$ in 3-dimensional space. Each step is randomly 1 unit forward (on x-axis), 1 unit back, 1 unit right (on y-axis), 1 unit left, 1 unit up (on z-axis), 1 unit down. On average, how far is he from the origin after 100 steps?
program random_walk
real :: distance, aver_dist = 0
integer :: w(3), direction(6, 3), randint
integer, parameter :: inf = 10**4

direction(1,:) = (/1,0,0/) ! forward one step
direction(2,:) = (/ -1,0,0/) ! back one step
direction(3,:) = (/0,1,0/) ! right one step
direction(4,:) = (/ _____ /)
direction(5,:) = (/ _____ /)
direction(6,:) = (/ _____ /)
do n=1, inf
   w = (/ _____ /) ! 3 dimensions, not two as in w=(/0,0/)
do i=1, 100
      w = w + direction( _____ ,:)
      ! direction(randint(1,4),:) is wrong
endo
distance = sqrt(real(sum(w**2)))
aver_dist = ____________________________
dondo
3 format ('Average distance after', a, f5.2)
print 3, ' 100 steps:', aver_dist
endprogram

integer function randint(m, n)
call random_number(r)
randint = floor(m+(n-m+1)*r)
endfunction ! Average distance after 100 steps is about 9.25