```
program generate_moon
   use, intrinsic :: iso_fortran_env, only: int32, real64
   implicit none

   integer(int32), parameter :: width = 1024
   integer(int32), parameter :: height = 768
   real(real64), parameter   :: pi = 3.14159265358979323846_real64

   character(len=32) :: loc_name(2)
   real(real64)      :: lat(2), lon(2), elev(2)
   integer(int32)    :: i

   loc_name(1) = "MINGA"
   lat(1) = 48.173333_real64
   lon(1) = 11.635000_real64
   elev(1) = 515.0_real64

   loc_name(2) = "GAGGERS"
   lat(2) = 48.333333_real64
   lon(2) = 11.250000_real64
   elev(2) = 500.0_real64

   do i = 1, 2
      call render_moon_image(trim(loc_name(i)), lat(i), lon(i), elev(i))
   end do

contains

   subroutine render_moon_image(location_name, latitude, longitude, elevation)
      character(len=*), intent(in) :: location_name
      real(real64), intent(in)     :: latitude, longitude, elevation

      character(len=64)   :: filename_ppm
      character(len=64)   :: filename_raw_png
      character(len=64)   :: filename_final_png
      character(len=4096) :: sys_cmd
      character(len=8)    :: dt_date, dt_time
      character(len=2048) :: overlay_text, overlay_text_sun
      character(len=128)  :: line_loc, line_zeit, line_phase
      character(len=128)  :: line_entf, line_az, line_alt
      character(len=128)  :: line_sun_az, line_sun_alt
      integer(int32)      :: img(width, height, 3)
      integer(int32)      :: x, y, cx, cy, radius, fov_radius
      real(real64)        :: dx, dy_up, dist, noise, brightness, wave

      real(real64)      :: alt, az, phase_pct, dist_km
      real(real64)      :: angle_sun_deg, angle_sun_rad
      real(real64)      :: az_sun, alt_sun
      real(real64)      :: u_sun_x, u_sun_y, u_perp_x, u_perp_y
      real(real64)      :: x_prime, y_prime, z_prime, k_phase
      real(real64)      :: sin_phi, cos_phi, dot_light
      real(real64)      :: scale_deg_per_px, alt_pixel
      character(len=32) :: phase_str
      integer           :: year, month, day, hour, minute, second

      filename_final_png = lowercase(location_name)//".png"
      filename_raw_png = lowercase(location_name)//"_raw.png"
      filename_ppm = lowercase(location_name)//".ppm"

      call date_and_time(date=dt_date, time=dt_time)
      read (dt_date(1:4), '(I4)') year
      read (dt_date(5:6), '(I2)') month
      read (dt_date(7:8), '(I2)') day
      read (dt_time(1:2), '(I2)') hour
      read (dt_time(3:4), '(I2)') minute
      read (dt_time(5:6), '(I2)') second

```

```
      write (sys_cmd, '(A,F9.6,A,F9.6,A,F6.1,A)') &
         "python3 /var/www/html/moon/get_moon.py ", &
         latitude, " ", longitude, " ", elevation, " > moon_tmp.txt"
      call execute_command_line(trim(sys_cmd))

      open (unit=20, file="moon_tmp.txt", status="old", action="read")
      read (20, *) az, alt, dist_km, phase_pct, phase_str, &
         angle_sun_deg, az_sun, alt_sun
      close (20)
      call execute_command_line("rm moon_tmp.txt")

      angle_sun_rad = angle_sun_deg*pi/180.0_real64
      u_sun_x = sin(angle_sun_rad)
      u_sun_y = cos(angle_sun_rad)
      u_perp_x = cos(angle_sun_rad)
      u_perp_y = -sin(angle_sun_rad)

      k_phase = max(0.001_real64, min(0.999_real64, phase_pct/100.0_real64))
      cos_phi = 2.0_real64*k_phase - 1.0_real64
      sin_phi = sqrt(1.0_real64 - cos_phi**2)

      ! Massstab: Sichtfeld-Durchmesser = 1.0 Grad (560 Pixel)
      ! Mond-Radius = 140 Pixel (Monddurchmesser 0.5 Grad)
      fov_radius = 280
      radius = 140
      scale_deg_per_px = 1.0_real64/(2.0_real64*real(fov_radius, real64))

      img = 0
      cx = width/2
      cy = height/2 + 30

      do y = 1, height
         dy_up = real(cy - y, real64)
         alt_pixel = alt + dy_up*scale_deg_per_px

         do x = 1, width
            dx = real(x - cx, real64)
            dist = sqrt(dx*dx + dy_up*dy_up)

            if (dist <= real(fov_radius, real64)) then
               if (alt_pixel < 0.0_real64) then
                  wave = sin(dx*0.1_real64)*cos(dy_up*0.2_real64)*4.0_real64
                  img(x, y, 1) = int(min(255.0_real64, &
                                         max(0.0_real64, &
                                             8.0_real64 + wave)))
                  img(x, y, 2) = int(min(255.0_real64, &
                                         max(0.0_real64, &
                                             28.0_real64 + wave*1.5_real64)))
                  img(x, y, 3) = int(min(255.0_real64, &
                                         max(0.0_real64, &
                                             55.0_real64 + wave*2.0_real64)))
               else
                  img(x, y, 1) = 12
                  img(x, y, 2) = 15
                  img(x, y, 3) = 22

                  if (dist <= real(radius, real64)) then
                     y_prime = dx*u_sun_x + dy_up*u_sun_y
                     x_prime = dx*u_perp_x + dy_up*u_perp_y

                     if (x_prime**2 + y_prime**2 &
                         <= real(radius, real64)**2) then
                        z_prime = sqrt(real(radius, real64)**2 &
                                       - x_prime**2 &
                                       - y_prime**2)
                        dot_light = (y_prime*sin_phi &
                                     + z_prime*cos_phi)/real(radius, real64)

```

```
                        if (dot_light > 0.0_real64) then
                           noise = sin(dx*0.15_real64)*cos(dy_up*0.15_real64) &
                                   + sin(dx*0.05_real64 + dy_up*0.08_real64)
                           brightness = (160.0_real64 &
                                         + noise*40.0_real64)*sqrt(dot_light)

                           img(x, y, 1) = int(min(255.0_real64, &
                                                  max(0.0_real64, brightness)))
                           img(x, y, 2) = int(min(255.0_real64, &
                                                  max(0.0_real64, brightness)))
                           img(x, y, 3) = int(min(240.0_real64, &
                                                  max(0.0_real64, &
                                                      brightness*0.95_real64)))
                        else
                           img(x, y, 1) = 18
                           img(x, y, 2) = 21
                           img(x, y, 3) = 30
                        end if
                     end if
                  end if
               end if
            end if
         end do
      end do

      open (unit=10, file=filename_ppm, status='replace', &
            action='write', form='formatted')
      write (10, '(A)') 'P3'
      write (10, '(I0, 1X, I0)') width, height
      write (10, '(A)') '255'
      do y = 1, height
         do x = 1, width
            write (10, '(I0, 1X, I0, 1X, I0)') &
               img(x, y, 1), img(x, y, 2), img(x, y, 3)
         end do
      end do
      close (10)

      sys_cmd = "pnmtopng "//trim(filename_ppm)//" &
&        > "//trim(filename_raw_png)//" && rm "//trim(filename_ppm)
      call execute_command_line(sys_cmd)

      write (line_loc, '(A,A,A,F9.6,A,F9.6,A)') &
         "STANDORT: ", trim(location_name), &
         " (", latitude, " N, ", longitude, " E)"

      write (line_zeit, '(A,I2.2,A,I2.2,A,I2.2,A,I2.2,A,I2.2,A,I4)') &
         "ZEIT: ", hour, ":", minute, ":", second, " CEST, ", &
         day, ".", month, ".", year

      write (line_phase, '(A,A,A,F5.1,A)') &
         "MONDPHASE: ", trim(phase_str), " (", phase_pct, "%)"

      write (line_entf, '(A,I6,A)') &
         "ENTF.: ", int(dist_km), " km"

      write (line_az, '(A,F5.1,A)') &
         "AZIMUT: ", az, " deg"

      write (line_alt, '(A,F5.1,A)') &
         "HOEHE: ", alt, " deg"

      overlay_text = "SIMULATION: MOND-ANSICHT ("// &
                     trim(location_name)//")"//char(10)//char(10)// &
                     trim(line_loc)//char(10)// &
                     trim(line_zeit)//char(10)// &
                     trim(line_phase)//char(10)// &
                     trim(line_entf)//char(10)// &
```

```
                    trim(line_az)//char(10)// &
                    trim(line_alt)//char(10)

      write (line_sun_az, '(A,F5.1,A)') "AZIMUT: ", az_sun, " deg"
      write (line_sun_alt, '(A,F5.1,A)') "HOEHE:  ", alt_sun, " deg"

      overlay_text_sun = "SONNE:"//char(10)// &
                         trim(line_sun_az)//char(10)// &
                         trim(line_sun_alt)

      sys_cmd = "convert "//trim(filename_raw_png)// &
                " -font /usr/share/fonts/truetype/dejavu/DejaVuSansMono.ttf"// &
                " -pointsize 16 -fill white"// &
                " -annotate +30+40 '"//trim(overlay_text)//"'"// &
                " -annotate +30+570 '"//trim(overlay_text_sun)//"' "// &
               trim(filename_final_png)//" && rm "//trim(filename_raw_png)

      call execute_command_line(sys_cmd)

   end subroutine render_moon_image

   pure function lowercase(str) result(res)
      character(len=*), intent(in) :: str
      character(len=len(str))      :: res
      integer                      :: j, code
      do j = 1, len(str)
         code = iachar(str(j:j))
         if (code >= 65 .and. code <= 90) then
            res(j:j) = achar(code + 32)
         else
            res(j:j) = str(j:j)
         end if
      end do
   end function lowercase

end program generate_moon
```