```
function surface_temperature, geometry, a, b, c, d, grid=grid, $
  longitude=longitude, latitude=latitude

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;;  Compute the surface temperature for an object as a function of
;;  latitude and longitude on the surface
;;
;;  Input either SZA, longitude/latitude, or neither
;;  If neither, then produces a map in latutude/longitude
;;
;;  Version 3.0: 12/15/2010
;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;; Figure out what inputs were given
nin = n_params()
case (nin) of
  1: begin
     lon0 = dindgen(361)*!dtor
     lat0 = dindgen(181)*!dtor-!pi/2.
     latlon = 1
     grid = 1
     end
  2: begin
     sza = a
     latlon = 0
     grid = 0
     end
  3: begin
     lon0 = a
     lat0 = b
     latlon = 1
     if (grid EQ !null) and (n_elements(a) NE n_elements(b)) then grid = 1
     if (grid EQ !null) then grid = 0
     end
  else: begin
    print, 'surface_temperature, geometry, grid=grid'
    print, 'surface_temperature, geometry, longitude, latitude, grid=1/0'
    print, 'surface_temperature, geometry, SZA'
    return, -1
    end
endcase

if (grid) then begin
  longitude = (lon0 # one(lat0))[*]
  latitude = (one(lon0) # lat0)[*]
endif

if (latlon) $
  then cosSZA = cos(longitude)*cos(latitude) $
```

```
52    else cosSZA = cos(SZA)
53
54  case strlowcase(geometry.startpoint) of
55    'mercury': begin
56      temp0 = 100.
57      temp1 = 600 + 125*(cos(geometry.taa)-1)/2. ;; sub-solar temp fn of taa
58      n = .25
59      day = where(cossza GT 0, nq)
60      temperature = replicate(temp0, n_elements(cossza))
61      if (nq GT 0) then temperature[day] += temp1*cosSZA[day]^n
62      end
63    else: stop
64  endcase
65
66  if (grid) then temperature = reform(temperature, n_elements(lon0), n_elements(lat0))
67
68  return, temperature
69
70  end
```