```
     ctl-opt dftactgrp(*no)     bnddir('WEBSERVICE');

    dcl-f weathrspfm  workstn(*ext) indds(dspindicators)
                  sfile(sfl:rrn);


     /copy /WebServices/weatherws/WeatherSoap.rpgleinc

    dcl-ds  WebServiceDS likeds(This_t);
    dcl-ds  ZipCodeDs    likeds(xsd_string);
    dcl-ds  ForecastDS   likeds(ForecastReturn_t);

     dcl-s rrn            packed(4:0);
     dcl-s workyear       packed(2:0);
     dcl-s workdays       packed(3:0);
     dcl-s workdatejul    packed(5:0);
     dcl-s zipcodealpha   char(5);
    // Since 'displaydate' is specified as a date without a datefmt AND
    // no special date format is specified on the ctl-opt spec
    // the default format is *iso
     dcl-s displaydate date;




     dcl-ds dspindicators;
     hourlyrequested    ind pos(1);
     endofjob           ind pos(3);
     // optional dcl-subf is required when a field name is the same as an op code
     dcl-subf return    ind pos(12);
     subfileclear       ind pos(60);
     subfiledsp         ind pos(62);
     zipcodezero        ind pos(71);
     zipcodenotfound    ind pos(72);
     webservicedown     ind pos(73);

     end-ds;

     //* Purpose: Launches www.weather.com with selected zip code
     //* Returns:
     dcl-pr displayhourly extpgm('HOURLYCL');
     zipcodealpha         char(5)  const;
     end-pr;



   /free

    //  Prompt user for zip code
       dou endofjob;

       exfmt screen1;
```

```
      //  Was EOJ requested?
          if endofjob;
           *inlr = *on;
           return;
          endif;

      //  Clear error conditons
          zipcodezero = *off;
          zipcodenotfound = *off;
          webservicedown = *off;

      //  Zipcode cannot be equal to all zeros
         if zipcode = *zeros;
          zipcodezero = *on;
          iter;
          endif;

      //  Prepare to call soap service
          clear WebServiceDs;
          clear ZipCodeDs;
          clear ForecastDs;

      //ZipCodeDs.value = %char(zipcode);
          ZipCodeDs.value = %editc(zipcode:'X');

      //  Allocate space in storage
           if stub_create_WeatherSoap(WebServiceDs);

       // Invoke the Web service operation.
      if stub_op_GetCityForecastByZip(WebServiceDs:ZipCodeDs:ForecastDs);
         endif;

       // Destroy Web service stubs.
        stub_destroy_WeatherSoap(WebServiceDs);
          endif;
       // Check if an error occurred during call to service
         if webserviceds.excoccurred or webserviceds.excstring > *blanks;
          webservicedown = *on;
           iter;
         endif;

      //  When here, we got some data!

      //  See if we have success (value of '1' will be returned)
          if forecastds.success.value = 0;
          zipcodenotfound = *on;
         iter;
       endif;

      //  If we made it this far, good data has been received
```

```
          city = forecastds.city.value;
          state = forecastds.state.value;
          wthrsta = forecastds.weatherstationcity.value;


        //  Prepare subfile control record
          rrn = *zeros;
          subfileclear = *on;
          subfiledsp = *off;

          write sflctl;

          subfileclear = *off;
          subfiledsp = *on;

        //  Capture detail data, load subfile
         for rrn = 1 to 7;
        //   wthrdate = %subst(weatherdata.forecast(rrn).date:1:10);
        // Description
         desc = forecastds.forecastresult.forecast...
         _ref.array(rrn).desciption.value;

        // Morning Low
         templo = forecastds.forecastresult.forecast...
         _ref.array(rrn).temperatures.morninglow.value;

        // Daytime High
         temphi = forecastds.forecastresult.forecast...
         _ref.array(rrn).temperatures.daytimehigh.value;

        // Precipitation Morning
         precpmo = forecastds.forecastresult.forecast...
         _ref.array(rrn).probabilityofprecipiatior.daytime.value;

        // Precipitation Nighttime
         precpnt = forecastds.forecastresult.forecast...
         _ref.array(rrn).probabilityofprecipiatior.nighttime.value;

        // Returned days will be days from January 1st (0-365)
         workdays  = forecastds.forecastresult.forecast...
         _ref.array(rrn).date.value.yday;

        // Returned year will be years from 1900, need to subtract 100
         workyear  = forecastds.forecastresult.forecast...
         _ref.array(rrn).date.value.year - 100;


        // Calculate julian work date.  If valid move to display date, else move *loval.
         workdatejul = (workyear * 1000) + workdays;
            test(de)  *jul workdatejul;
            if not %error;
              displaydate = %date(workdatejul:*jul);
```

```
            wthrdate = %char(displaydate);
          else;
            wthrdate = '01-01-0001';
          endif;

         write sfl;
         endfor;

       //  Display subfile.  If F1 is selected display weather.com
          dou return;
           exfmt sflctl;
             if hourlyrequestec;
               zipcodealpha = %editc(zipcode:'X');
               displayhourly(zipcodealpha);
             endif;
          enddo;
            enddo;
```