Prueba de Caja Blanca

"Seguimiento a graduados IASA-I"

Integrantes:

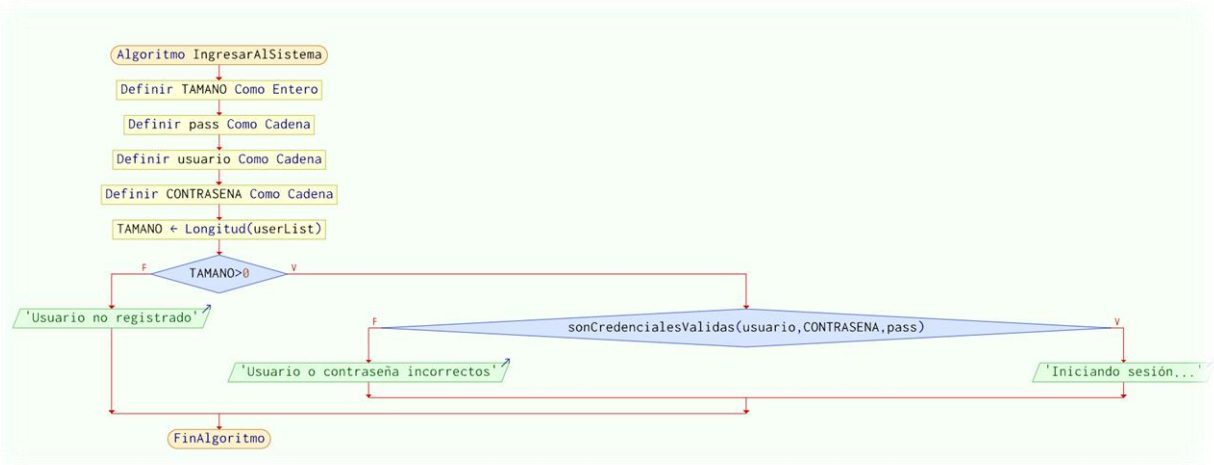
Genesis Calapaqui
Alex Paguay
Santiago Sañay

Fecha 2024-02-18

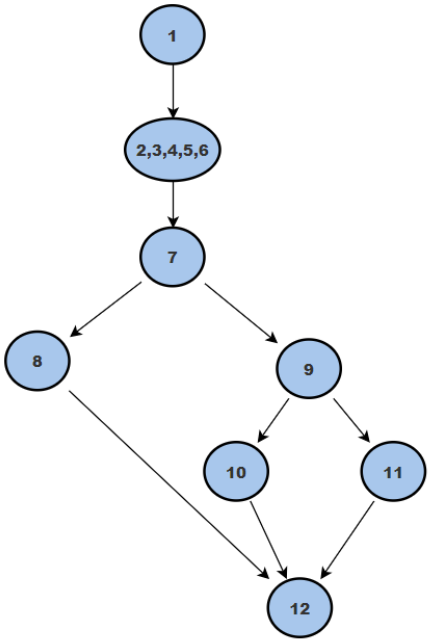
Req. 02: INGRESAR AL SISTEMA
CÓDIGO FUENTE

```
if (size>0){
    String pass=userList.get(0).getPassword();
    if (isValidCredentials(username, password,pass)) {
        Common.setUsername(userList.get(0).getUsername());
        Common.setPassword(userList.get(0).getPassword());
        Intent intent = new Intent(LoginActivity.this,
MainActivity.class);
        startActivity(intent);
        finish();
        errorTextView.setVisibility(View.GONE); // Ocultar el mensaje
de error si estaba visible
    } else {
        errorTextView.setVisibility(View.VISIBLE);
        errorTextView.setText("Usuario o contraseña incorrectos");
    }

}else{
    errorTextView.setVisibility(View.VISIBLE);
    errorTextView.setText("Usuario o contraseña incorrectos");
}
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4,5,6,7,8,12
R2: 1, 2, 3, 4,5,6,7,9,10,12
R3: 1, 2, 3, 4,5,6,7,9,11,12

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

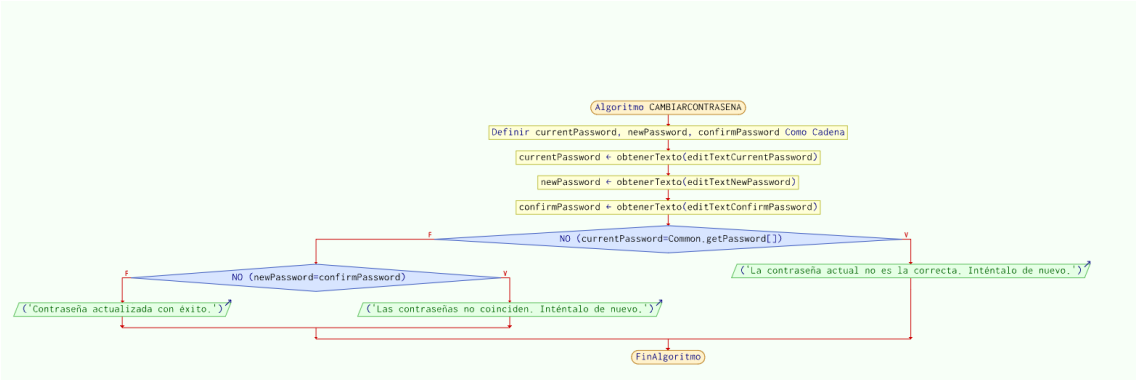
- V(G) = número de nodos predicados(decisiones)+1
  V(G)=2+1=3
- V(G) = A – N + 2
  V(G)= 9– 8 + 2 = 3

DONDE:
P: Número de nodos predicado
A: Número de aristas
N: Número de nodos

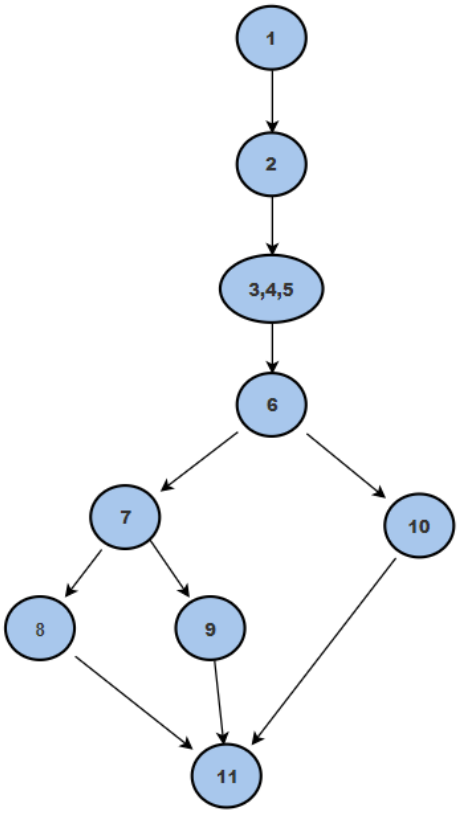
Req. 03: CAMBIAR CONTRASEÑA
CÓDIGO FUENTE

```
private boolean updatePassword() {
    String currentPassword =
editTextCurrentPassword.getText().toString();
    String newPassword = editTextNewPassword.getText().toString();
    String confirmPassword =
editTextConfirmPassword.getText().toString();

    // Validar si las contraseñas coinciden
    if (!currentPassword.equals(Common.getPassword())) {

        Toast.makeText(this, "La contraseña actual no es la correcta.
Inténtalo de nuevo.", Toast.LENGTH_SHORT).show();

        return false;
    }
    if (!newPassword.equals(confirmPassword)) {

        Toast.makeText(this, "Las contraseñas no coinciden. Inténtalo
de nuevo.", Toast.LENGTH_SHORT).show();

        return false;
    }

        // Mostrar un mensaje de éxito si la contraseña se actualizó
correctamente.
    registerPassword();
    Toast.makeText(this, "Contraseña actualizada con éxito.",
Toast.LENGTH_SHORT).show();
    return true;
}
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4,5,6,7,8,11
R2: 1, 2, 3, 4,5,6,7,9,11
R3: 1, 2, 3, 4,5,6,10,11

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)=2+1=3
- V(G) = A – N + 2
  V(G)= 10– 9 + 2 = 3

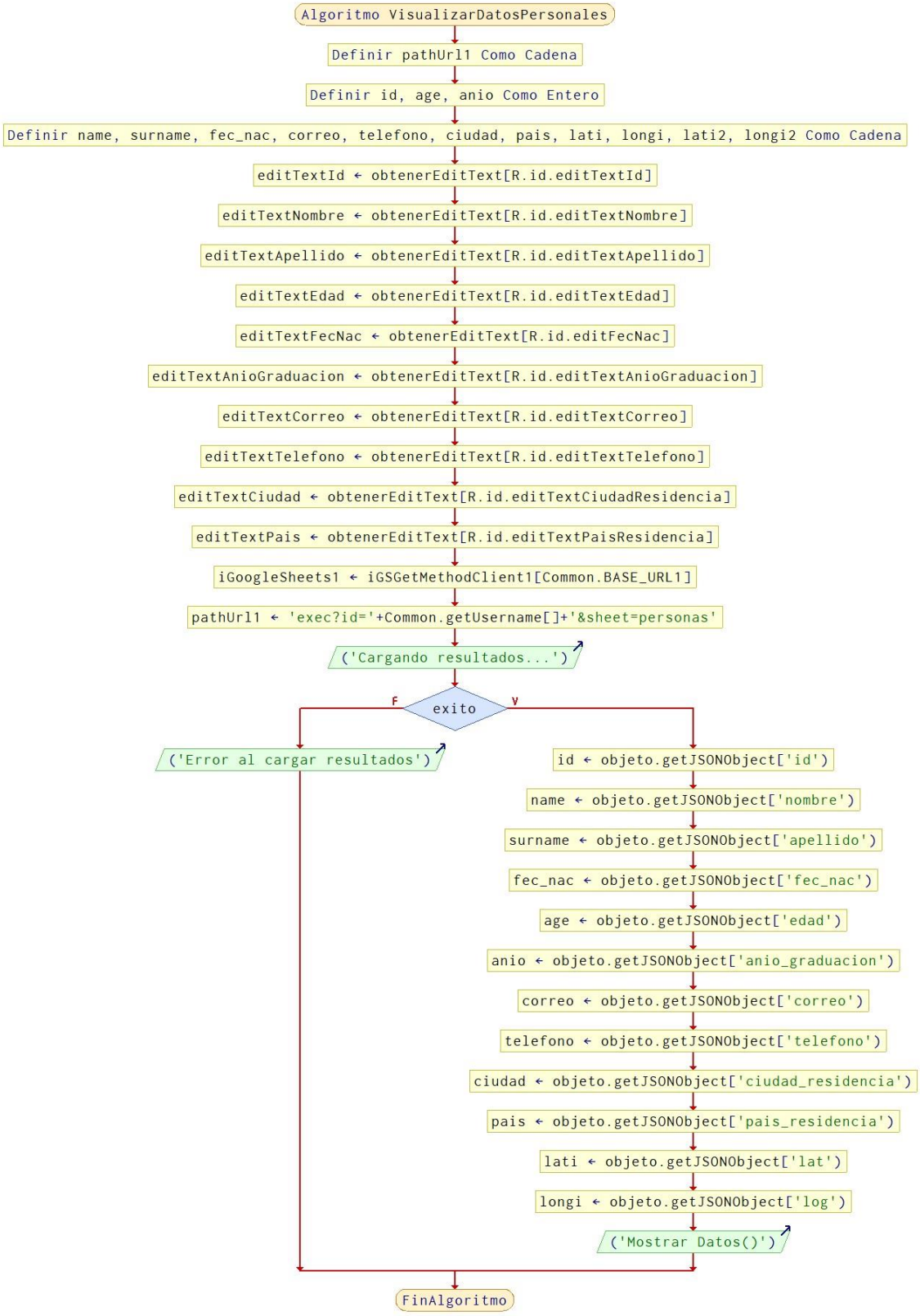
DONDE:
P: Número de nodos predicado
A: Número de aristas
N: Número de nodos

Req. 04: VISUALIZAR DATOS PERSONALES
CÓDIGO FUENTE

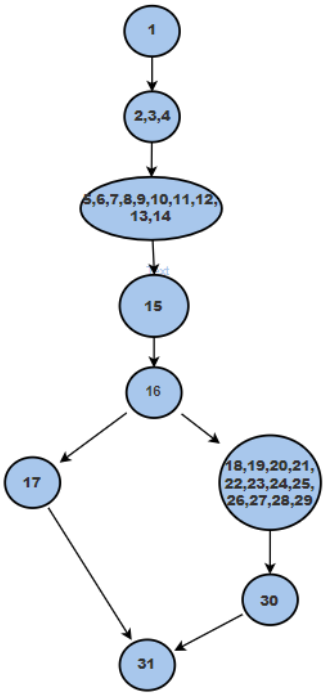
```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_user_details);

    // Asigna los EditTexts a las variables de la clase
    editTextId = findViewById(R.id.editTextId);

    editTextNombre = findViewById(R.id.editTextNombre);
    editTextApellido = findViewById(R.id.editTextApellido);
    editTextEdad = findViewById(R.id.editTextEdad);
    editTextFecNac = findViewById(R.id.editFecNac);
    editTextAnioGraduacion=findViewById(R.id.editTextAnioGraduacion);
    editTextCorreo=findViewById(R.id.editTextCorreo);
    editTextTelefono=findViewById(R.id.editTextTelefono);
    editTextCiudad=findViewById(R.id.editTextCiudadResidencia);
    editTextPais=findViewById(R.id.editTextPaisResidencia);

    iGoogleSheets1 = Common.iGSGetMethodClient1(Common.BASE_URL1);
    String pathUrl1;
    pathUrl1 = "exec?id=" +
Common.getUsername().toString()+"&sheet=personas";

    String pathUrl;
    progressDialog = ProgressDialog.show(UserDetailsActivity.this,
            "Cargando resultados",
            "Espere por favor",
            true,
            false);
    try {
        //editTextId.setText(pathUrl1);

        iGoogleSheets1.getPeople(pathUrl1).enqueue(new
Callback<String>() {

            public void onResponse(@NonNull Call<String> call,
@NonNull Response<String> response) {
                try {

                    assert response.body() != null;
                    JSONObject responseObject = new
JSONObject(response.body());
                    JSONArray peopleArray =
responseObject.getJSONArray("persons");


//editTextId.setText(String.valueOf(peopleArray.length()));

                    JSONObject object = peopleArray.getJSONObject(0);

                    String id = object.getString("id");

                    String name = object.getString("nombre");

                    String surname = object.getString("apellido");
                    String fec_nac=object.getString("fec_nac");
                    String age = object.getString("edad");
                    String anio = object.getString("anio_graduacion");
                    String correo = object.getString("correo");
                    String telefono= object.getString("telefono");

                    String ciudad =
```

```
object.getString("ciudad_residencia");
                    String pais = object.getString("pais_residencia");

                    String lati=object.getString("lat");
                    String longi=object.getString("log");
                    String lati2=lati.replace(",",".");
                    String longi2=longi.replace(",",".");

                    double latitud = Double.parseDouble(lati2);
                    double longitud = Double.parseDouble(longi2);

                    Common.setLat(latitud);
                    Common.setLog(longitud);







                    editTextId.setText(id);
                    editTextNombre.setText(name);
                    editTextApellido.setText(surname);
                    editTextFecNac.setText(fec_nac);
                    editTextEdad.setText(age);
                    editTextAnioGraduacion.setText(anio);
                    editTextCorreo.setText(correo);
                    editTextTelefono.setText(telefono);
                    editTextCiudad.setText(ciudad);
                    editTextPais.setText(pais);

                    onMapReady(mMap);
                    progressDialog.dismiss();



                } catch (JSONException je) {
                    je.printStackTrace();
                }
            }

            @Override
            public void onFailure(@NonNull Call<String> call, @NonNull
Throwable t) {

            }
        });
    } catch (Exception e) {
        e.printStackTrace();
    }


    // Asigna el Listener al botón de regreso
    buttonBack = findViewById(R.id.buttonBack);
    buttonBack.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            // Regresar al MainActivity
            finish();
        }
    });
}
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4,5,6,7,8,9,10,11,12,13,14,15,16,17,31
R2: 1, 2, 3, 4,5,6,7,8,9,10,11,12,13,14,15,16,18,19,20,21,22,23,24,25,26,27,28,29,30,31

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

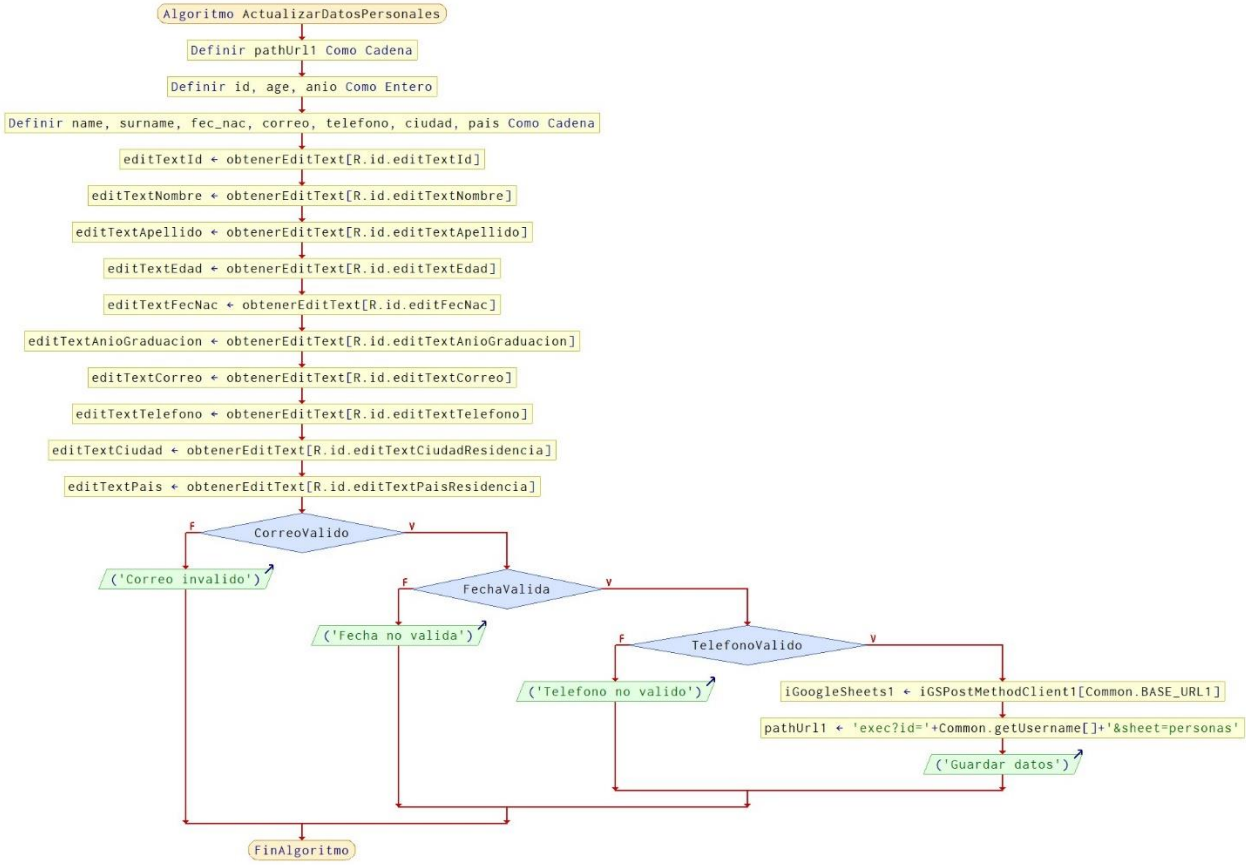
- V(G) = número de nodos predicados(decisiones)+1
  V(G)=1+1=2
- V(G) = A – N + 2
  V(G)= 9 – 9 + 2 = 2

DONDE:
P: Número de nodos predicado
A: Número de aristas
N: Número de nodos

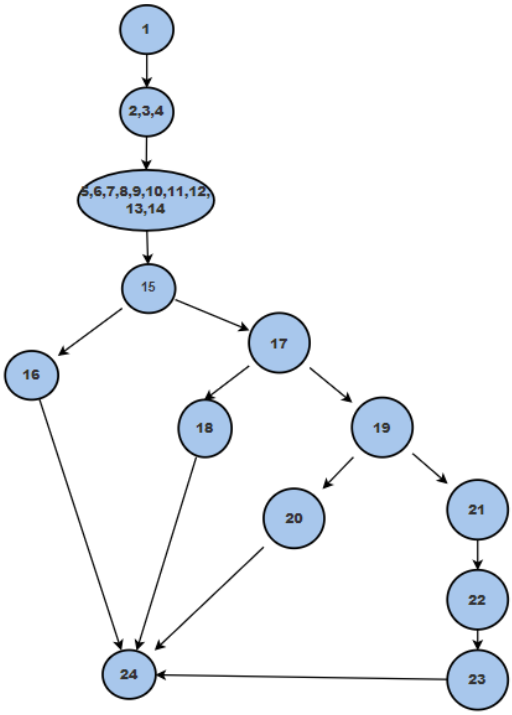
Req. 05: ACTUALIZAR DATOS PERSONALES
CÓDIGO FUENTE

```
    public void afterTextChanged(Editable editable) {
        // Validar si el texto ingresado es una dirección de correo
electrónico válida
        boolean isEmailValid = isValidEmail(editable.toString());

        if (isEmailValid) {
            // El correo electrónico es válido, limpiamos el error si
estaba presente
            editTextCorreo.setError(null);
        } else {
            // El correo electrónico no es válido, mostramos un
mensaje de error
            editTextCorreo.setError("Correo electrónico inválido");
        }
        // Habilitar o deshabilitar el botón según la validez del
correo y el teléfono
        enableSaveButton(isEmailValid,
isValidPhone(editTextTelefono.getText().toString()));
    }
});

// Agregar un TextWatcher al EditText de teléfono para validar el
teléfono en tiempo real
editTextTelefono.addTextChangedListener(new TextWatcher() {
    @Override
    public void beforeTextChanged(CharSequence charSequence, int i,
int i1, int i2) {
    }

    @Override
    public void onTextChanged(CharSequence charSequence, int i, int
i1, int i2) {
    }

    @Override
    public void afterTextChanged(Editable editable) {
        // Validar si el texto ingresado es un teléfono válido
        boolean isPhoneValid = isValidPhone(editable.toString());

        if (isPhoneValid) {
            // El teléfono es válido, limpiamos el error si estaba
presente
            editTextTelefono.setError(null);
        } else {
            // El teléfono no es válido, mostramos un mensaje de error
            editTextTelefono.setError("Teléfono inválido");
        }
        // Habilitar o deshabilitar el botón según la validez del
correo y el teléfono

enableSaveButton(isValidEmail(editTextCorreo.getText().toString()),
isPhoneValid);
    }
});
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4,5,6,7,8,9,10,11,12,13,14,15,16,24
R2: 1, 2, 3, 4,5,6,7,8,9,10,11,12,13,14,15,17,18,24
R3: 1, 2, 3, 4,5,6,7,8,9,10,11,12,13,14,15,17,19,20,24
R4: 1, 2, 3, 4,5,6,7,8,9,10,11,12,13,14,15,17,19,21,22,23,24

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

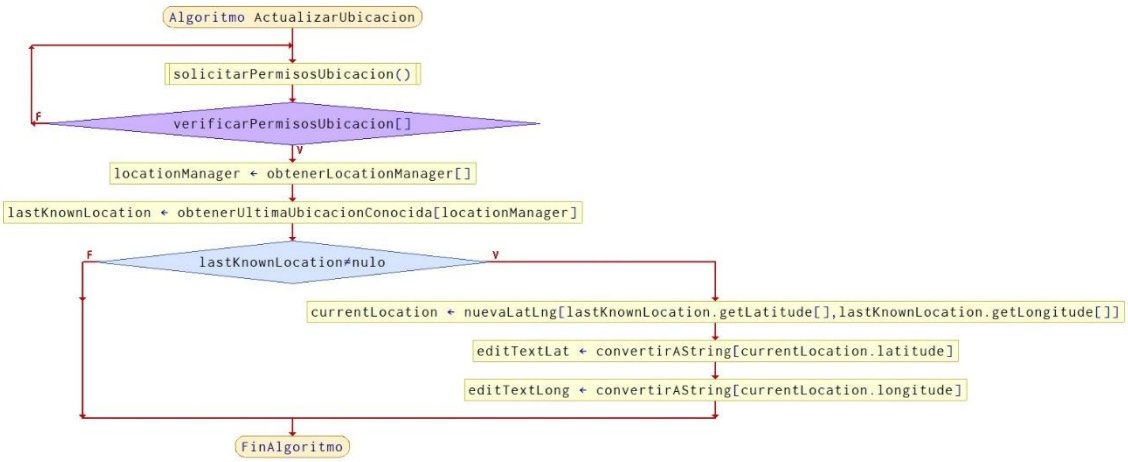
- V(G) = número de nodos predicados(decisiones)+1
  V(G)=3+1=4
- V(G) = A – N + 2
  V(G)= 15 – 13 + 2 = 4

DONDE:
P: Número de nodos predicado
A: Número de aristas
N: Número de nodos

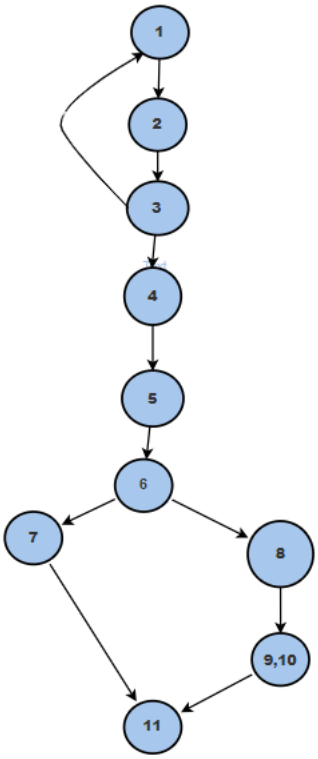
Req. 06: ACTUALIZAR UBICACIÓN GEOGRÁFICA
CÓDIGO FUENTE

```
    private void centerMapOnMyLocation() {

    do{
// Solicitar permisos de ubicación si no están otorgados
        ActivityCompat.requestPermissions(this,
                new
String[]{Manifest.permission.ACCESS_FINE_LOCATION},

    }while(ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION)
            != PackageManager.PERMISSION_GRANTED);
    // Verificar permisos de ubicación

        // Obtener la ubicación actual
        LocationManager locationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
        Location lastKnownLocation =
locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER);
        if (lastKnownLocation != null) {
            LatLng currentLocation = new
LatLng(lastKnownLocation.getLatitude(),
lastKnownLocation.getLongitude());

mMap.animateCamera(CameraUpdateFactory.newLatLngZoom(currentLocation,
15));

editTextLat.setText(String.valueOf(currentLocation.latitude));

editTextLong.setText(String.valueOf(currentLocation.longitude));
        }

}
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 1
R2: 1, 2, 3, 4,5,6,7,11
R3: 1, 2, 3, 4,5,6,8,9,10,11

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)=2+1=3
- V(G) = A – N + 2
  V(G)= 11 – 10+ 2 = 3

DONDE:
P: Número de nodos predicado
A: Número de aristas
N: Número de nodos