

Rutas de monitoreo y metadatos (R)

Lina María Quintero Fonseca

2026-02-21

Tabla de contenidos

| | | |
|-----|--|---|
| 0.1 | Ejercicio 1 | 4 |
| 0.2 | Ejercicio 2 (R: c() vs list()) | 5 |
| 0.3 | Pregunta general (Slicing) | 5 |

httpgd::hgd(host = "0.0.0.0", port = 8787, token = FALSE) httpgd::hgd_details() plot(1:10)

```
# Ejercicio 1: Rutas de monitoreo

punto_1 <- c(11.1198, -74.0321)
punto_2 <- c(11.1250, -74.0280)
punto_3 <- c(11.1302, -74.0215)

ruta_avistamiento <- list(punto_1, punto_2, punto_3)

punto_4 <- c(11.1355, -74.0150)
ruta_avistamiento <- append(ruta_avistamiento, list(punto_4))

ultimo_reporte <- ruta_avistamiento[[length(ruta_avistamiento)]]

elevaciones <- c(850.5, 920.0, 1050.2, 1180.8)

elevacion_maxima <- max(elevaciones)
total_puntos <- length(ruta_avistamiento)

cat("Elevación máxima:", elevacion_maxima, "m\n")
```

Elevación máxima: 1180.8 m

```
cat("Total de puntos (vértices):", total_puntos, "\n")
```

Total de puntos (vértices): 4

```
cat("Último reporte (Lat, Lon):", ultimo_reporte[1], ultimo_reporte[2], "\n")
```

Último reporte (Lat, Lon): 11.1355 -74.015

```
# Ejercicio 2: Metadatos y Ecosistemas (R)
```

```
pnn_tayrona <- list(  
  nombre = "PNN Tayrona",  
  area_hectareas = 15000,  
  abierto_turismo = TRUE,  
  ecosistemas = c("Manglar", "Bosque Seco", "Coral")  
)  
  
# Acceso seguro a "fecha_creacion"  
fecha_creacion <- if ("fecha_creacion" %in% names(pnn_tayrona)) {  
  pnn_tayrona$fecha_creacion  
} else {  
  "Dato no disponible"  
}  
print(fecha_creacion)
```

[1] "Dato no disponible"

```
# Agregar nueva clave  
pnn_tayrona$departamento <- "Magdalena"  
  
# Quitar duplicados  
observaciones_crudas <- c("Manglar", "Coral", "Bosque Seco", "Manglar", "Coral", "Matorral")  
observaciones_unicas <- unique(observaciones_crudas)  
  
# Intersección con ecosistemas oficiales  
ecosistemas_avistados_hoy <- intersect(pnn_tayrona$ecosistemas, observaciones_unicas)  
  
print(observaciones_unicas)
```

[1] "Manglar" "Coral" "Bosque Seco" "Matorral"

```
print(ecosistemas_avistados_hoy)
```

```
[1] "Manglar"      "Bosque Seco" "Coral"
```

```
# rutas_pnn.py
# Ejercicio 1: Rutas de monitoreo (Tuplas y Listas)

# 1) Puntos como tuplas (inmutables)
punto_1 = (11.1198, -74.0321)
punto_2 = (11.1250, -74.0280)
punto_3 = (11.1302, -74.0215)

# 2) Ruta como lista (mutable y secuencial)
ruta_avistamiento = [punto_1, punto_2, punto_3]

# 3) Agregar cuarto punto al final
punto_4 = (11.1355, -74.0150)
ruta_avistamiento.append(punto_4)

# 4) Extraer último punto con indexación negativa
ultimo_reporte = ruta_avistamiento[-1]

# 5) Lista paralela de elevaciones
elevaciones = [850.5, 920.0, 1050.2, 1180.8]

# 6) Calcular e imprimir elevación máxima y total de puntos
elevacion_max = max(elevaciones)
total_puntos = len(ruta_avistamiento)

print("Último reporte:", ultimo_reporte)
```

```
Último reporte: (11.1355, -74.015)
```

```
print("Elevación máxima:", elevacion_max)
```

```
Elevación máxima: 1180.8
```

```
print("Total de puntos visitados:", total_puntos)
```

```
Total de puntos visitados: 4
```

```
# Ejercicio 2: Metadatos y Ecosistemas (Diccionarios y Conjuntos)

# 1) Diccionario con claves y valores exactos
pnn_tayrona = {
    "nombre": "PNN Tayrona",
    "area_hectareas": 15000,
    "abierto_turismo": True,
    "ecosistemas": ["Manglar", "Bosque Seco", "Coral"],
}

# 2) Acceso seguro a clave inexistente
fecha_creacion = pnn_tayrona.get("fecha_creacion", "Dato no disponible")
print("Fecha de creación:", fecha_creacion)
```

Fecha de creación: Dato no disponible

```
# 3) Actualizar diccionario
pnn_tayrona["departamento"] = "Magdalena"

# 4) Eliminar duplicados con set
observaciones_crudas = ["Manglar", "Coral", "Bosque Seco", "Manglar", "Coral", "Matorral"]
observaciones_unicas = set(observaciones_crudas)

# 5) Intersección: ecosistemas oficiales vs observados hoy
ecosistemas_oficiales = set(pnn_tayrona["ecosistemas"])
ecosistemas_avistados_hoy = ecosistemas_oficiales.intersection(observaciones_unicas)

print("Observaciones únicas:", observaciones_unicas)
```

Observaciones únicas: {'Matorral', 'Manglar', 'Bosque Seco', 'Coral'}

```
print("Ecosistemas oficiales avistados hoy:", ecosistemas_avistados_hoy)
```

Ecosistemas oficiales avistados hoy: {'Manglar', 'Bosque Seco', 'Coral'}

0.1 Ejercicio 1

En términos de seguridad del código, guardar cada punto como un objeto separado evita que se modifique accidentalmente por operaciones sobre la ruta completa. La ruta se mantiene como lista mutable para poder crecer con nuevos reportes. Si se intentara “cambiar” el valor de un punto, en R

se podría reasignar el vector, pero la buena práctica es tratar los puntos como registros inmutables y crear uno nuevo, no editarlo en sitio.

0.2 Ejercicio 2 (R: `c()` vs `list()`)

Usaría `c()` para crear el vector de ecosistemas (`c("Manglar", "Bosque Seco", "Coral")`) porque es un vector atómico de caracteres. Y lo almacenaría como campo dentro de una lista nombrada (propiedades) porque el objeto completo mezcla tipos (texto, numérico, lógico, vector), y `list()` mantiene esa estructura heterogénea sin simplificarla.

0.3 Pregunta general (Slicing)

En Python, `ruta[1:3]` devuelve 2 elementos porque el límite superior es exclusivo. En R, `ruta[1:3]` devuelve 3 elementos porque el rango 1:3 incluye ambos extremos.

Python (`ruta[1:3]`): el extremo derecho no se incluye (intervalo semiabierto). Devuelve elementos con índices 1 y 2 \rightarrow 2 elementos. Motivo: la convención de slicing en Python es [inicio, fin).

R (`ruta[1:3]`) y Julia (`ruta[1:3]`): el rango sí incluye ambos extremos. Devuelve índices 1, 2 y 3 \rightarrow 3 elementos. Motivo: en R y Julia, 1:3 construye un vector/rango inclusivo.