```
library(nloptr)



# Problem 1

f1 <- function(x) {

return(-x[1] * x[2]) # Objective function: -xy

}

constraints1 <- function(x) {

return(c(x[1] + x[2]^2 - 2, -x[1], -x[2])) # Constraints: x + y^2 <= 2, x >= 0, y >= 0

}

result1 <- nloptr(x0 = c(1, 1), eval_f = f1, eval_g_ineq = constraints1,

                  opts = list(algorithm = "NLOPT_GN_ISRES", xtol_rel = 1e-6))
```

```
print(result1)



# Problem 2

f2 <- function(x) {

return(2 * x[1] + x[2]) # Objective function: 2x + y

}

constraints2 <- function(x) {

return(c(sqrt(x[1]^2 + x[2]^2) - 2, -x[1], (0.5 * x[1] - 1) - x[2])) # Corrected Constraints: sqrt(x^2 + y^2) <= 2, x >= 0, y <= 0.5x - 1

}

result2 <- nloptr(x0 = c(1, 1), eval_f = f2, eval_g_ineq = constraints2,

                  opts = list(algorithm = "NLOPT_GN_ISRES", xtol_rel = 1e-6))
```

```
print(result2)



# Problem 3

f3 <- function(x) {

return(sum(x^2)) # Objective function: x1^2 + x2^2 + x3^2 + x4^2

}

constraints3 <- function(x, A) {

return(c(sum(x) - 1, A - x[4])) # Corrected Constraints: x1 + x2 + x3 + x4 = 1, only x4 >= A

}



A_values <- c(0.25, 0.25, 0.5) # Cases for A: < 1/4 (A=0.25), = 1/4 (A=0.25), > 1/4 (A=0.5)
```

```
results3 <- list()

for (A in A_values) {

result <- nloptr(x0 = rep(0.25, 4), eval_f = f3,

                 eval_g_eq = function(x) { return(sum(x) - 1) }, # Equality constraint: sum of variables equals 1

                 eval_g_ineq = function(x) { return(A - x[4]) }, # Corrected inequality constraint: only x4 >= A

                 opts = list(algorithm = "NLOPT_GN_ISRES", xtol_rel = 1e-6))

results3[[paste("A =", A)]] <- result

}

print(results3)
```