11.Critical Path Method

1. Construct the network, find the critical path and project duration of the following activity

Activity:	**1-2**	**1-3**	**2-4**	**2-5**	**3-4**	**4-5**
Duration:	8	4	10	2	5	3

```
# Load required library
library(igraph)
# Define the activities
activities <- c("1", "2", "3", "4", "5")
# Define the edges (dependencies)
edges <- matrix(c(1, 2, 1, 3, 2, 4, 2, 5, 3, 4, 4, 5),
                ncol = 2, byrow = TRUE)
# Define the durations for each edge
edge_durations <- c(8, 4, 10, 2, 5, 3)
# Create the graph
g <- graph_from_edgelist(edges, directed = TRUE)
# Set edge attributes for duration
E(g)$duration <- edge_durations
# Plot the graph
plot(g, layout = layout_nicely(g), vertex.label = activities, vertex.size = 30)
# Find all possible paths from Node 1 to Node 5
possible_paths <- all_simple_paths(g, from = 1, to = 5)
# Function to calculate the value of each path
calculate_path_value <- function(path) {
  total_duration <- 0
  for (i in 1:(length(path) - 1)) {
    edge <- c(path[i], path[i + 1])
    total_duration <- total_duration + edge_durations[which(edges[,1] == edge[1] &
edges[,2] == edge[2])]
```

```
  }
  return(total_duration)
}
# Calculate the value for each path
path_values <- sapply(possible_paths, calculate_path_value)
# Combine paths and their values
paths_and_values <- data.frame(Path = sapply(possible_paths, function(path)
paste(activities[path], collapse = " -> ")), Value = path_values)
# Print paths and their values
print(paths_and_values)
# Find the index of the critical path (path with maximum value)
critical_path_index <- which.max(path_values)
# Extract the critical path
critical_path <- possible_paths[[critical_path_index]]
# Calculate the value (duration) of the critical path
critical_path_value <- path_values[critical_path_index]
# Convert the critical path to a readable format
critical_path_readable <- paste(activities[critical_path], collapse = " -> ")
# Print the critical path and its value
print(paste("Critical Path:", critical_path_readable))
print(paste("Critical Path Value (Duration):", critical_path_value))
```

12.PERT Analysis

Construct the network, find the expected time duration, expected variance, critical path and project length of the following activity

Activity:	1-2	1-3	2-4	3-4	4-5	3-5
a(optimistic):	2	9	5	2	6	8
m(most_likely):	5	12	14	5	6	17
b(pessimistic):	14	15	17	8	12	20

```
# Load required library

library(igraph)

# Define the activities and their durations

activities <- c("1-2", "1-3", "2-4", "3-4", "4-5", "3-5")

duration_a <- c(2, 9, 5, 2, 6, 8)

duration_m <- c(5, 12, 14, 5, 6, 17)

duration_b <- c(14, 15, 17, 8, 12, 20)

# Compute expected time using the formula

expected_time <- (duration_a + 4 * duration_m + duration_b) / 6

# Compute variance using the formula

variance <- ((duration_b - duration_a) / 6)^2

# Create a table for computations

computations <- data.frame(Activities = activities, Expected_Time = expected_time,
Variance = variance)

# Print the table

print("Tabulated Computations for Expected Time and Variance:")

print(computations)

# Define the activities and their durations

activities <- c("1", "2", "3", "4", "5")

# Define the edges (dependencies)

edges <- matrix(c(1, 2, 1, 3, 2, 4, 3, 4, 4, 5,3,5),
```

```
            ncol = 2, byrow = TRUE)
# Define the durations for each edge
#edge_durations <- c(6, 12, 13, 5, 7, 16)
edge_durations<-expected_time
# Create the graph
g <- graph_from_edgelist(edges, directed = TRUE)
# Plot the graph
plot(g, layout = layout_nicely(g), vertex.label = activities, vertex.size = 50)
# Find all possible paths from Node 1 to Node 5
possible_paths <- all_simple_paths(g, from = 1, to = 5)
# Function to calculate the value of each path
calculate_path_value <- function(path) {
  total_duration <- 0
  for (i in 1:(length(path) - 1)) {
    edge <- c(path[i], path[i + 1])
    total_duration <- total_duration + edge_durations[which(edges[,1] == edge[1] &
edges[,2] == edge[2])]
  }
  return(total_duration)
}
# Calculate the value for each path
path_values <- sapply(possible_paths, calculate_path_value)
# Combine paths and their values
paths_and_values <- data.frame(Path = sapply(possible_paths, function(path)
paste(activities[path], collapse = " -> ")), Value = path_values)
# Print paths and their values
print(paths_and_values)
# Find the index of the critical path (path with maximum value)
```

```
critical_path_index <- which.max(path_values)
# Extract the critical path
critical_path <- possible_paths[[critical_path_index]]
# Calculate the value (duration) of the critical path
critical_path_value <- path_values[critical_path_index]
# Convert the critical path to a readable format
critical_path_readable <- paste(activities[critical_path], collapse = " -> ")
# Print the critical path and its value
print(paste("Critical Path:", critical_path_readable))
print(paste("Critical Path Value (Duration):", critical_path_value))
```