APPENDICES

APPENDIX-A: Python program for Design of Box Inlet Spillway

```
import math

g = 9.81

Qg = float(input("\nRunoff collected from catchment area (meter cube per second)
Qg = "))
B = float(input("\nEnter the value of B (m) = "))
W = float(input("\nEnter the value of width at the upstream end = "))
We = float(input("\nEnter the value of width at the end sill = "))

# critical depth at straight section
x1 = Qg / W
x2 = x1 ** 2
x3 = x2 / g
x4 = math.sqrt(x3)
Dc = x4
print("\nCritical depth at straight section, Dc =", Dc)

# critical depth at end sill
x1 = Qg / We
x2 = x1 ** 2
x3 = x2 / g
x4 = math.sqrt(x3)
Dce = x4
print("\nCritical depth at end sill, Dce =", Dce)

# Length of straight section
Ls = ((0.2 / (B / W)) + 1) * Dc
print("\nLength of straight section, Ls =", Ls)

# Length of basin section
Lb = ((W + 2 * B) / (2 * B / W))
print("\nLength of basin section, Lb =", Lb)

# Tail water depth
if Lb > 11.5 * Dce:
    D2 = (Dce + (0.052 * We))
else:
    D2 = 1.6 * Dce
print("\nTail Water Depth, d2 =", D2)
```

```
# Height of end sill and longitudinal sill
F = D2 / 6
print("\nHeight of end sill and longitudinal sill, F =", F)

# Number of longitudinal sills
N = 2 if We < 2.5 * W else 4
print("\nNumber of longitudinal Sills :", N)

# Space between longitudinal sills
P1 = W / 6
P2 = W / 4
print("\nSpacing between longitudinal sills, P =", P1, "to", P2)

# Height of side wall above tail water Depth
t = D2 / 3
print("\nHeight of side wall above tail water Depth, t =", t)
```

Code Break Down:

1. **Constants and Inputs**:
    - **g = 9.81**: This defines the acceleration due to gravity in meters per second squared.
    - The user is prompted to input several parameters:
        - **Qg**: Runoff collected from a catchment area in cubic meters per second.
        - **B**: The value of a parameter **B** in meters.
        - **W**: The value of the width at the upstream end in meters.
        - **We**: The value of the width at the end sill in meters.
2. **Calculating Critical Depths**:
    - Critical depth at the straight section (**Dc**) and end sill (**Dce**) are calculated using specific formulas. These calculations involve the input values and the gravitational constant.
    - The **math. sqrt()** function is used to calculate the square root. This function is from the **math** module, so the code starts with **import math**.
3. **Length of Sections**:
    - Lengths of the straight section (**Ls**) and basin section (**Lb**) are computed using given formulas. These calculations involve **B**, **W**, and the critical depths.
4. **Tail Water Depth**:
    - Depending on the condition of **Lb**, the tailwater depth (**D2**) is computed. It involves **Dce** and **We**.
5. **Height of End Sill and Longitudinal Sill (F)**:
    - The height of the end sill and longitudinal sill (**F**) is calculated using **D2**.
6. **Number of Longitudinal Sills (N)**:
    - Depending on the condition of **We**, the number of longitudinal sills (**N**) is determined.
7. **Spacing Between Longitudinal Sills (P)**:
    - The spacing between longitudinal sills (**P**) is calculated based on the value of **W**.
8. **Height of Side Wall Above Tail Water Depth (t)**:
    - The height of the side wall above the tailwater depth (**t**) is calculated using **D2**.

The code is essentially performing a series of calculations based on user inputs and predefined formulas to determine various parameters related to hydraulic structures.

APPENDIX-B: Python program for Design of Straight Drop Spillway

```
import math

def calculate_runoff():
    ans = input("\nDo you have the coefficient of runoff (Y/N): ")
    if ans.lower() == 'y':
        c = float(input("\nEnter the coefficient of runoff: "))
    elif ans.lower() == 'n':
        no = int(input("\nEnter the number of different sections of the watershed
                       (maximum limit 5): "))
        coef = []
        area = []
        ca = 0
        total_area = 0
        for flag in range(no):
            coef_flag = float(input(f"\nEnter the coefficient of runoff (c{flag}): "))
            area_flag = float(input(f"Enter the area of the section (a{flag}): "))
            coef.append(coef_flag)
            area.append(area_flag)
            ca += coef_flag * area_flag
            total_area += area_flag
        c = ca / total_area
        print("\n\nThe overall coefficient of runoff is:", c)
    else:
        print("\nInvalid entry")
        return c

    ch = input("\nDo you have the rainfall intensity (Y/N): ")
    if ch.lower() == 'y':
        i = float(input("\nEnter the rainfall intensity (cm/hr): "))
    elif ch.lower() == 'n':
        length_tc = float(input("\nEnter the length of the runoff contributing area (m): "))
        slope_tc = float(input("Enter the slope of that area (in decimal, i.e.,
                       percentage/100): "))
        l_tc = length_tc ** 0.77
        s_tc = slope_tc ** -0.385
        tc = 0.01947 * l_tc * s_tc
        print("\nThe time of concentration is:", tc, "min")

        return_period = float(input("\nEnter the value for return period (year): "))
        imperical_a = float(input("Enter the value for imperical_a (unitless): "))
        imperical_b = float(input("Enter the value for imperical_b (unitless): "))
        imperical_K = float(input("Enter the value for imperical_K (unitless): "))
        imperical_e = float(input("Enter the value for imperical_e (unitless): "))

        intermediate_i1 = return_period ** imperical_a
```

```
        intermediate_i2 = imperical_K * intermediate_i1
        intermediate_i3 = ((tc / 60) + imperical_b)
        intermediate_i4 = intermediate_i3 ** imperical_e
        intermediate_i5 = intermediate_i2 / intermediate_i4
        i = intermediate_i5
        print("\nRainfall Intensity is =", i, "(cm/hr)")
    else:
        print("\nInvalid entry")
        return i

    A = float(input("\nEnter the catchment area (hectares): "))
    print("\nc =", c, ", i =", i, ", A =", A)

    Qg = c * i * A / 36
    print("\nRunoff collected from catchment area Qg (cubic meters per second) =", Qg)

    return Qg

def design_spillway(Qg):
    print("\nConsidering different lengths (meters) 3, 3.5, 4, 4.5, 5; and selecting the
                suitable length according to the hydrological design")

    f = float(input("\nEnter the drop of bed (meters): "))

    # Calculations for the Height of structure
    l = 3
    while l <= 5:
        x = (Qg * (1.1 + 0.01 * f)) / (1.711 * l)
        h = x ** 0.666667
        if (h / f) < 0.5:
            print("\nHeight (meters) =", h)
            break
        l += 0.5

    # Calculation of Minimum length of Head Wall Extensions
    E1 = (3 * h) + 0.6
    E2 = 1.5 * f
    E = E1 if E1 > E2 else E2
    print("\nThe minimum length of Head wall extensions (meters) =", E)

    # Calculation of Length of basin
    Lb = f * (2.28 * (h / f) + 0.52)
    print("\nThe Length of basin (meters) =", Lb)

    # Calculation of Height of Longitudinal Wall
    S = h / 4
    print("\nThe Height of Longitudinal Wall (meters) =", S)
```

```
    # Calculation of Height of wing wall and side wall at junction
    J1 = 2 * h
    J2 = (f + S + h - (Lb + 0.1 / 2))
    J = J1 if J1 > J2 else J2
    print("\nThe Height of Wing Wall and side wall at junction (meters) =", J)

    # Calculation of end sill and transverse sill
    Ht = h / 3
    print("\nHeight of end sill and transverse sill (meters) =", Ht)

    # Components
    M = 2 * (f + 1.33 * h - J)
    K = (Lb + 0.1) - M
    print("\nComponents M and K (meters): M =", M, "and K =", K)

def main():
    Qg = calculate_runoff()
    design_spillway(Qg)

if __name__ == "__main__":
    main()
```

Code Break Down:

1. **Importing Math Module**:
    - **import math**: This line imports the **math** module, which provides mathematical functions.
2. **Function calculate_runoff()**:
    - This function calculates the runoff collected from a catchment area based on user inputs.
    - It first asks whether the coefficient of runoff is known (**Y/N**).
    - If the coefficient is known (**Y**), it prompts the user to input it.
    - If not, it asks for the number of different sections of the watershed and calculates the overall coefficient of runoff.
    - Then, it asks whether rainfall intensity is known (**Y/N**).
    - If known (**Y**), it prompts the user to input it.
    - If not, it calculates the rainfall intensity based on various inputs such as the time of concentration and return period.
    - Finally, it calculates the runoff using the formula **Qg = c * i * A / 36**, where **c** is the coefficient of runoff, **i** is the rainfall intensity, and **A** is the catchment area.
3. **Function design_spillway(Qg)**:
    - This function designs the spillway based on the runoff calculated by **calculate_runoff()**.
    - It considers different lengths for the spillway and selects the suitable length according to hydrological design criteria.
    - Then, it calculates various parameters such as the height of the structure, minimum length of head wall extensions, length of the basin, the height of the longitudinal wall, height of wing wall and side wall at the junction, height of end sill and transverse sill, and components **M** and **K**.
4. **Function main()**:
    - This function serves as the entry point of the program.
    - It calls **calculate_runoff()** to calculate the runoff and then passes the result to **design_spillway()** to design the spillway.
5. **Execution**:
    - Finally, the **main()** function is called if the script is executed directly (**__name__ == "__main__"**).

Overall, this code allows users to calculate runoff and design a spillway based on their input parameters.

APPENDIX-C: Python program for Design of Pipe Spillway

```
import math

g = 9.81
Pi = 3.14

c = float(input("Enter the coefficient of runoff: "))
i = float(input("Enter the rainfall intensity (cm/hr): "))
A = float(input("Enter the catchment area (hectares): "))
print(f"c = {c}, i = {i}, A = {A}")

Qp = c * i * A / 36
# 50% of peak flow is required to discharge of pipe
Q = 0.5 * Qp

print(f"\nDischarge (m3/s), Q = {Q}")

h = float(input("Enter the drop (m): "))
Ke = float(input("Entrance loss coefficient (ke): "))
Kc = float(input("Friction loss coefficient (kc): "))
l = float(input("Enter the length(m): "))  # l > BW
print(f"h = {h}, Ke = {Ke}, Kc = {Kc}, l = {l}")

X1 = 2 * g * h
X2 = math.sqrt(X1)
X3 = 1 + Ke + Kc * l
X4 = math.sqrt(X3)
X5 = Q * X4 * 4 / (Pi * X2)
d = math.sqrt(X5)
print(f"\nDiameter of pipe, d = {d}(meter)")

V = Q * 4 / (Pi * d * d)
print(f"Velocity of flow, V = {V}")

X1 = Kc * V * V / (2 * g)
X2 = 1 - (X1 * X1)
X3 = math.sqrt(X2)
Sn = X1 / X3
print(f"Natural slope Sn = {Sn}")

print("\nAssuming the elevation difference, dH = 0.3")
dH = 0.3
S = dH / l
print(f"The conduit slope S = {S}")

if S < Sn:
```

```
    print("Full flow condition")
else:
    print("Partial flow condition")
```

Code Break Down:

1. **Importing Math Module**: This line imports a module named **math**, which provides access to various mathematical functions and constants.
2. **Constants**: Here, the script sets some constant values:
    - **g**: Represents the acceleration due to gravity, which is 9.81 m/s^2.
    - **Pi**: Represents the mathematical constant Pi, which is approximately 3.14.
3. **User Inputs**: The script prompts the user to input certain parameters:
    - **c**: Coefficient of runoff.
    - **i**: Rainfall intensity in centimetres per hour.
    - **A**: Catchment area in hectares. These inputs are essential forthe calculation.
4. **Calculating Peak Discharge**: It computes the peak discharge (**Qp**) based on the user inputs and then calculates half of it (**Q**) because only 50% of the peak flow is required to discharge from the pipe.
5. **User Inputs for Pipe Characteristics**: The script prompts for more inputs related to the pipe system:
    - **h**: Drop (height difference) in meters.
    - **Ke**: Entrance loss coefficient.
    - **Kc**: Friction loss coefficient.
    - **l**: Length of the pipe in meters.
6. **Pipe Diameter Calculation**: Using the inputs provided, the script calculates the diameter of the pipe (**d**) using a series of formulas.
7. **Flow Velocity Calculation**: It computes the flow velocity (**V**) based on the discharge and pipe diameter.
8. **Calculating Natural Slope and Conduit Slope**: The script calculates the natural slope (**Sn**) and conduit slope (**S**) based on certain formulas using the provided parameters.
9. **Determining Flow Condition**: Finally, based on the comparison between the conduit slope and the natural slope, the script determines whether it's a full flow condition or a partial flow condition and prints the result.

APPENDIX-D: Python program for Design of Chute Spillway

```
# Import math module 5.10 question
import math

# Define constants
g = 9.81
Pi = 3.14

# Input the coefficient of runoff
c = float(input("Enter the coefficient of runoff : "))

# Input the rainfall intensity (cm/hr)
i = float(input("Enter the rainfall intensity (cm/hr) : "))

# Input the catchment area (hactares)
A = float(input("Enter the catchment area (hactares) : "))

print(f"c= {c}, i= {i}, A= {A}")

# Calculate the peak discharge (m3/s)
Qp = c * i * A / 36

print(f"\nPeak Discharge (m3/s), Qp= {Qp}")

# Input the depth over the flow over weir (meter)
H = float(input("\nEnter depth over the flow over weir (meter) : "))

# Calculate the length of weir
X1 = math.pow(H, 1.5)
L = Qp / (1.77 * X1)

print(f"\nLength of weir: {L}")

# Input the value of drop
h = float(input("\nEnter the value of drop, h = "))

# Calculate the height of fall
Hf = h - (0.1 * h)

# Calculate the velocity of flow
X1 = 2 * g * Hf
V = math.sqrt(X1)

# Calculate the initial depth
d1 = Qp / (L * V)
print(f"\n Initial depth d1, = {d1}")
```

```
# Calculate the Froud number
X1 = g * d1
X2 = math.sqrt(X1)
Fr1 = V / X2

print(f"\n Froud Number 1, Fr1= {Fr1}")

# Calculate the sequent depth
X1 = 1 + (8 * Fr1 * Fr1)
X2 = math.sqrt(X1)
X3 = X2 - 1
d2 = X3 * d1 / 2

print(f"\n Sequent depth d2, = {d2}")

# Calculate the height of chute/ floor block
print(f"\n Height of chute/ floor block : {d1}")

# Calculate the width and spacing between chute/floor block
X1 = 0.75 * d1
print(f"\n Width and spacing between chute/floor block : {X1}")

# Calculate the basin length
X2 = math.pow(Fr1, 0.38)
Lb = 4.5 * d2 / X2
print(f"\n Basin length : {Lb}")

# Calculate the spacing of floor block from upstream of stilling basin
X3 = Lb / 3
print(f"\n Spacing of floor block from upstream of stilling basin : {X3}")

# Calculate the minimum distance of floor block from side wall
X4 = 3 * d1 / 8
print(f"\n Minimum distance of floor block from side wall : {X4}")

# Calculate the height of end sill
X5 = 0.07 * d2
print(f"\n Height of end sill : {X5}")

# Calculate the actual tail water depth
X6 = math.pow(Fr1, 0.45)
D2 = 1.4 * X6 * d1
print(f"\n Actual tail water depth : {D2}")

# Calculate the height of side wall and wing wall at junction
J = (d2 / 3) + D2
print(f"\n Height of side wall and wing wall at junction : {J}")
```

```
# Calculate the freeboard
FB = d2 / 3
print(f"\n Freeboard = {FB}")
0
```

Code Break Down:

1. **Importing Math Module**: The code starts by importing the math module to use mathematical functions later in the script.
2. **Defining Constants**: Constants like the acceleration due to gravity (**g**) and the value of Pi (**Pi**) are defined.
3. **User Inputs**: The script prompts the user to input parameters like:
    - **c**: Coefficient of runoff.
    - **i**: Rainfall intensity in centimetres per hour.
    - **A**: Catchment area in hectares. These inputs are crucial for the hydraulic calculations.
4. **Calculating Peak Discharge**: It computes the peak discharge (**Qp**) using the provided formula and prints it.
5. **Calculating Length of Weir**: The script computes the length of the weir based on the depth over the flow over the weir (**H**), using a formula, and prints it.
6. **Calculating Hydraulic Parameters**: It calculates various hydraulic parameters like initial depth (**d1**), Froude number (**Fr1**), sequent depth (**d2**), height of chute/floor block, width and spacing between chute/floor block, basin length, spacing of floor block from upstream of stilling basin, minimum distance of floor block from side wall, height of end sill, actual tail water depth, height of side wall and wing wall at junction, and freeboard using respective formulas, and prints each of them.