```python
import numpy as np
from math import factorial

def coefficientMatrix(num_equation):
    coefMatrix = np.zeros((num_equation, num_equation))
    for q in range(num_equation):
        for j in range(num_equation):
            coefMatrix[q, j] = np.power(j, q) / factorial(q)
    return coefMatrix

def Adams(r, name):
    alpha = np.zeros(r+1)
    beta = np.zeros(r+1)
    alpha[r] = 1
    alpha[r-1] = -1
    num_equations = r + 1
    if "Bashforth" in name:
        num_equations -= 1
    B = coefficientMatrix(num_equations)
    A = np.array([(np.power(r, q+1) - np.power(r-1, q+1)) /
factorial(q+1) for q in range(num_equations)])
    beta[:num_equations] = np.linalg.solve(B, A)
    return alpha, beta

### Backward Differentiation Formulas
def BackwardDifferentiation(r):
    A = coefficientMatrix(r+1)
    B = np.array([0.] + [np.power(r, i-1) / factorial(i-1) for i in
range(1, r+1)]) # we assume that betta_r value is one
    alpha = np.linalg.solve(A, B)
    beta = np.zeros(r+1)
    beta[-1] = 1
    return alpha, beta

def AlphaBetaCoefficients(r: int, method: str):
    if method == "Backward Differentiation":
        alpha, beta = BackwardDifferentiation(r)
    elif method in ["Adams-Bashforth", "Adams-Moulton"]:
        alpha, beta = Adams(r, method)
    else:
        raise "Wrong input method"
    return alpha, beta

def printCoefficients(alpha, beta):
    # Print the header
    print(f"{'Index':<6} {'Alpha':<10} {'Beta':<10}")
    print("-" * 30)

    # Print the values in column format
    for i, (a, b) in enumerate(zip(alpha, beta)):
        print(f"{i:<6} {a:<10.5f} {b:<10.5f}")

print("Adams-Bashforth Method")
for r in range(1, 5):
    alpha, beta = AlphaBetaCoefficients(r, "Adams-Bashforth")
    printCoefficients(alpha, beta)
```

```
Adams-Bashforth Method
Index  Alpha      Beta
------------------------------
0      -1.00000   1.00000
1      1.00000    0.00000
Index  Alpha      Beta
------------------------------
0      0.00000    -0.50000
1      -1.00000   1.50000
2      1.00000    0.00000
Index  Alpha      Beta
------------------------------
0      0.00000    0.41667
1      0.00000    -1.33333
2      -1.00000   1.91667
3      1.00000    0.00000
Index  Alpha      Beta
------------------------------
0      0.00000    -0.37500
1      0.00000    1.54167
2      0.00000    -2.45833
3      -1.00000   2.29167
4      1.00000    0.00000
```

```python
print("Adams-Moulton Method")
for r in range(1, 5):
    alpha, beta = AlphaBetaCoefficients(r, "Adams-Moulton")
    printCoefficients(alpha, beta)
```

```
Adams-Moulton Method
Index  Alpha      Beta
------------------------------
0      -1.00000   0.50000
1      1.00000    0.50000
Index  Alpha      Beta
------------------------------
0      0.00000    -0.08333
1      -1.00000   0.66667
2      1.00000    0.41667
Index  Alpha      Beta
------------------------------
0      0.00000    0.04167
```

```
1      0.00000    -0.20833
2      -1.00000   0.79167
3      1.00000    0.37500
Index  Alpha      Beta
------------------------------
0      0.00000    -0.02639
1      0.00000    0.14722
2      0.00000    -0.36667
3      -1.00000   0.89722
4      1.00000    0.34861
```

```
print("Backward Differentiation Formulas Method")
for r in range(1, 5):
    alpha, beta = AlphaBetaCoefficients(r, "Backward Differentiation")
    printCoefficients(alpha, beta)
```

```
Backward Differentiation Formulas Method
Index  Alpha      Beta
------------------------------
0      -1.00000   0.00000
1      1.00000    1.00000
Index  Alpha      Beta
------------------------------
0      0.50000    0.00000
1      -2.00000   0.00000
2      1.50000    1.00000
Index  Alpha      Beta
------------------------------
0      -0.33333   0.00000
1      1.50000    0.00000
2      -3.00000   0.00000
3      1.83333    1.00000
Index  Alpha      Beta
------------------------------
0      0.25000    0.00000
1      -1.33333   0.00000
2      3.00000    0.00000
3      -4.00000   0.00000
4      2.08333    1.00000
```