[ ]:
```julia
h = 0.5
a = 1
b = 4

steps = Int(ceil((b - a) / h)) + 1
X_vec = a:h:b
Y_vec = zeros(steps)

for i in 1:steps
    Y_vec[i] = acos(log2(X_vec[i]/2))
end

print(" x  |  acos(log2(x/2))\n")
print("-----------------------\n")
for i in 1:steps
    print(X_vec[i], " | ", Y_vec[i], "\n")
end
```

```
 x  |  acos(log2(x/2))
-----------------------
1.0 | 3.141592653589793
1.5 | 1.9987803462433056
2.0 | 1.5707963267948966
2.5 | 1.2430310324200673
3.0 | 0.9459625046862551
3.5 | 0.6311407394425016
4.0 | 0.0
```

[ ]:
```julia
differences = zeros(steps, steps)

for i in 1:steps
    differences[i, 1] = Y_vec[i]
end

for i in 2:steps
    for j in 1:(steps-i+1)
        differences[j, i] = differences[j+1, i-1] - differences[j, 
 ↪i-1]
    end
end

print("Конечные разности: \n\n")
for i in 1:steps
    for j in 1:steps
        print(round(differences[i, j], digits=3), "\t")
    end
    print("\n")
```

```
end
```

```
Разделенные разности:

3.142   -1.143  0.715   -0.615  0.545   -0.524  0.253
1.999   -0.428  0.1     -0.07   0.021   -0.271  0.0
1.571   -0.328  0.031   -0.048  -0.25   0.0     0.0
1.243   -0.297  -0.018  -0.299  0.0     0.0     0.0
0.946   -0.315  -0.316  0.0     0.0     0.0     0.0
0.631   -0.631  0.0     0.0     0.0     0.0     0.0
0.0     0.0     0.0     0.0     0.0     0.0     0.0
```

[ ]:
```
function NewtonFixPolyForward(X_vec, differences, x, h)
    size = length(X_vec)
    q = (x - X_vec[1]) / h
    polynom = 1.0
    answer = differences[1, 1]

    for col in 2:size
        row = 1

        polynom *= (q - col + 2)
        answer += differences[row, col] * polynom / factorial(col - 1)
    end
    return answer
end



x1 = 1.32
print("x1 = ", x1, "\n\n")
print("       Pn(1)(x1) = ", NewtonFixPolyForward(X_vec, differences,␣
  ↪x1, h), "\n")
print("acos(log2(x1/2)) = ", acos(log2(x1/2)), "\n")
```

```
x1 = 1.32

       Pn(1)(x1) = 2.2642976149315883
acos(log2(x1/2)) = 2.2136251930241557
```

[ ]:
```
function NewtonFixPolyBackward(X_vec, differences, x, h)
    size = length(X_vec)
    q = (x - X_vec[size]) / h
    polynom = 1.0
    answer = differences[size, 1]

    for col in 2:size
        row = size - col + 1
```

```
        polynom *= (q + col - 2)
        answer += differences[row, col] * polynom / factorial(col - 1)
    end
    return answer
end

x2 = 3.84
print("x2 = ", x2, "\n\n")
print("       Pn(2)(x2) = ", NewtonFixPolyBackward(X_vec,
  ↪differences, x2, h), "\n")
print("acos(log2(x2/2)) = ", acos(log2(x2/2)), "\n")
```

```
x2 = 3.84

       Pn(2)(x2) = 0.26700449041324326
acos(log2(x2/2)) = 0.344908744643400054
```