0: sin

1: summSin

2: summSinNechet

massPQ[n-1] + Math.sin((2\*n-1)\*fi);

3: summSinDiv

massPQ[n-1] - (Math.sin(n\*fi)/constanta); massPQ[n-1] + (Math.sin(n\*fi)/constanta);

4: sinProd

5 \* Math.sin((n+1)\*fi) \* Math.sin(n\*fi);

5: summCos

massPQ[n-1] + Math.cos(n\*fi);

6: summSinZnakoPeremen

massPQ[n-1] - Math.sin(n\*fi); massPQ[n-1] + Math.sin(n\*fi);

7: summCosZnakoPeremen

massPQ[n-1] - Math.cos(n\*fi); massPQ[n-1] + Math.cos(n\*fi);

8: cosRaznost

2 \* (Math.cos(fi) - Math.cos((2\*n+1)\*fi) );

9: sinDiv

Math.sin((n+1)\*fi) / Math.sin(n\*fi);

10: summSinNechetZnakoPeremen

massPQ[n-1] - Math.sin((2\*n-1)\*fi); massPQ[n-1] + Math.sin((2\*n-1)\*fi);

11: summCosNechetZnakoPeremen

massPQ[n-1] - Math.cos((2\*n-1)\*fi); massPQ[n-1] + Math.cos((2\*n-1)\*fi);

14: summSinChetnZnakoPeremen

massPQ[n] = massPQ[n-1] - Math.sin((2\*n)\*fi); else massPQ[n-1] + Math.sin((2\*n)\*fi);

15: summCosChetnZnakoPeremen

massPQ[n] = massPQ[n-1] - Math.cos((2\*n)\*fi); else massPQ[n-1] + Math.cos((2\*n)\*fi);

16: summSin4n\_1

massPQ[n] = massPQ[n-1] + Math.sin((4\*n-3)\*fi);

18: summCosZnakoPeremen\_DivSin

massPQ[n] = (massPQ2[n-1] + Math.cos((2\*n-1)\*fi)) / (this.massPQ3[n-1] + Math.sin((2\*n-1)\*fi))

19: summCosZnakoPeremen\_DivSin

massPQ[n] = (massPQ2[n-1] + Math.cos(n\*fi) / (this.massPQ3[n-1] + Math.sin(n\*fi)

20: summChetnSinDivNecetSin

massPQ[n] = (massPQ2[n-1] + Math.sin(2\*n\*fi) / (this.massPQ3[n-1] + Math.((2\*n-1)\*fi)

21: summChetnSinDivNecetSin

massPQ[n] = (massPQ2[n-1] + Math.sin(2\*n\*fi) / (this.massPQ3[n-1] + Math.((2\*n-1)\*fi)

22: summ\_N\_Sin

massPQ[n] = massPQ[n-1] + (1+1/n)\*Math.sin(n\*fi);

26: cos\_sin\_arctan(n, fi, a)

massPQ[n] = Math.sqrt(a\*a +fi\*fi)\*( Math.sin((n+1)\*Math.atan(fi/a))/Math.sin(n \* Math.atan(fi/a)) ) - a;

27: cos\_Drob(n, pq1, pqCh, pqZ)

massPQ[n] =

if (i2 === i) { this.massPQ2[i2] = pqCh/pqZ;

} else { this.massPQ2[i2] = pqCh/(pqZ-this.massPQ2[i2+1]); }

if (i2 === 1) { this.massPQ[n] = pq1 - this.massPQ2[1]; }

28: cos\_Drob(n, pq1, pqCh, pqZ)

massPQ[n] =

if (i2 === i) { this.massPQ2[i2] = pqCh/pqZ;

} else { this.massPQ2[i2] = pqCh/(pqZ-this.massPQ2[i2+1]); }

if (i2 === 1) { this.massPQ[n] = pq1 - this.massPQ2[1]; }

29: cos\_sin\_arctan(n, fi, a)

massPQ[n] = Math.sqrt(a\*a +fi\*fi)\*( Math.sin((n+1)\*Math.atan(fi/a))/Math.sin(n \* Math.atan(fi/a)) ) - a;

30: ch\_Sin\_arctg(n, fi, a)

this.massPQ[n] =Math.sqrt(a\*a +fi\*fi)\*(Math.sin((n+1)\*Math.atan(fi/a))/Math.sin(n\*Math.atan(fi/a)) )-a;

if (Math.cosh(this.massPQ[n]) > 1000000) {

this.massPQ[n] = 1000000

} else {

this.massPQ[n] = Math.cosh(this.massPQ[n]);

}

31: sh\_Sin\_arctg(n, fi, a)

this.massPQ[n] =Math.sqrt(a\*a +fi\*fi)\*(Math.sin((n+1)\*Math.atan(fi/a))/Math.sin(n\*Math.atan(fi/a)) )-a;

if (Math.sinh(this.massPQ[n]) > 1000000) {

this.massPQ[n] = 1000000

} else {

this.massPQ[n] = Math.sinh(this.massPQ[n]);

}

32: exp\_Sin\_arctg(n, fi, a)

this.massPQ[n] =Math.sqrt(a\*a +fi\*fi)\*(Math.sin((n+1)\*Math.atan(fi/a))/Math.sin(n\*Math.atan(fi/a)) )-a;

if (Math.exp(this.massPQ[n]) > 1000000) {

this.massPQ[n] = 1000000

} else {

this.massPQ[n] = Math.exp(this.massPQ[n]);

}

38: Drob\_smal\_th(n, fi, a)

let pq1 = a, pqCh = a\*a + fi\*fi;

if (n === 1) {

this.massPQ2[n] = pq1;

} else {

this.massPQ2[n] = pq1 - (pqCh/(pq1+this.massPQ2[n-1]));

}

return this.massPQ[n] = Math.tanh(this.massPQ2[n]);

},