Some Looooops

Example I: 2 Stopping Criteria

```
...
declaration and initialization of variables;

while (Math.abs(delta) >= TOL && iter <= 500){
    iter += 1;
    f     =   Math.pow(x,4) + ...;
    deriv = 4*Math.pow(x,3) + ...;
    delta = -f/deriv;
    x += delta;
}
Format.print(System.out,’’The root is %8.4f ’’,x)
Format.print(System.out,’’after %3d iterations.\n’’,iter)
```

Example II: Nested while-loops for finding appropriate increment size

```
...
// check=0 : Root has not been found
// check=1 : Root has been found
int    check=0;
double inc=0.0001;

// decrease increment until increment is small enough
// to find a root with LHS-method
while (inc>=.00001 && check==0){
    double x=3.3;
    int iter=0;
    double f_x= C1* Math.pow(x,4) + C3*Math.pow(x,2) + C4*x + C5;

    // check stopping criterium and upper bound for x
    while (Math.abs(f_x)>TOL && x<=4){
        x += inc;
        f_x= C1* Math.pow(x,4) + C3*Math.pow(x,2) + C4*x + C5;
        iter=iter+1;
    } // 2nd while-loop

    // root has been found if gotten out of 2nd while-loop before
    // x reached 4! Assign 1 to check to get out of 1st while-loop
    if (x<4){
        Format.print(System.out,"The root, %7.5f, was found.\n",x);
        check=1;
    }
    else
    // root has not been found! Hence, decrease increment.
        inc -= 0.00001;
} // 1st while-loop
```

Example III: Nested for-loop and while-loop for determining the root for different tolerances

```
...
int stop = 5;          // number of times to decrease order of magnitude of
double tolerance = 1.;

for(int i=1;i<=stop;++i) {

    // Use Newton's Method
    do {
        f =  Math.pow(root,4)+ ...;
        fp = 4*Math.pow(root,3)+ ...;
        root = root - f/fp;
    }   while(Math.abs(f) > tolerance);

    root = root + f/fp; // if succeeded, restore root to previous value

    // report current solution of root for given tolerance
    Format.print(System.out, "| %6.3f ",tolerance);
    Format.print(System.out, "| %12.10f ",root);
    Format.print(System.out, "| %12.10f |\n",f);

    // pick tolerance -- lower by an order of magnitude each loop
    tolerance = tolerance/10;

} // end for
```