```
sweepRs (generic function with 1 method)
```

```
#########sweep particle radius and output force matrix to file##########
function sweepRs(parPosTmp::Matrix{Float64},Rs::Float64)
parPos=deepcopy(parPosTmp);
dRs=0.0001;
println("Input Data file name and press Enter:");
fileName=readline();
outputData=open(fileName*".txt","w");
totRndNo=30;
rndDigNo=9;
for iTmp in 1:totRndNo
write(outputData,"round"*string(iTmp));
println("round"*string(iTmp));
write(outputData,"\n");
write(outputData,"Rs: "*string(Rs));
write(outputData,"\n");
write(outputData,"initial positions: ");
write(outputData,"\n");
writedlm(outputData,parPos,'\t');
write(outputData,"\n");
parPos=md(parPos,Rs);
write(outputData,"equilibrium positions: ");
write(outputData,"\n");
writedlm(outputData,parPos,'\t');
write(outputData,"\n");
kMat=forceMat(parPos,Rs);
write(outputData,"force matrix: ");
write(outputData,"\n");
writedlm(outputData,kMat,'\t');
println(kMat);
write(outputData,"\n");
egv=eigen(kMat);
write(outputData,"eigenvalues: ");
write(outputData,"\n");
writedlm(outputData,round.(egv.values;digits=rndDigNo),'\t');
println(round.(egv.values,digits=rndDigNo));
write(outputData,"\n");
write(outputData,"eigenvectors: ");
write(outputData,"\n");
writedlm(outputData,round.(egv.vectors;digits=rndDigNo),'\t');
write(outputData,"\n");
write(outputData,"\n");
Rs=Rs+dRs;
flush(outputData);
end
close(outputData);
return 0;
end
```

```
sweepRsS (generic function with 1 method)
```

```
######sweep particle radius and output force matrix to file##########
function
sweepRsS(center::Bool,pNo::Integer,foldNo::Integer,ringNo::Integer,Rc::Array{Float64}
,Rs::Float64)
dRs=0.0001;
println("Input Data file name and press Enter:");
fileName=readline();
outputData=open(fileName*".txt","w");
totRndNo=30;
rndDigNo=12;
rcTmp=Rc;
parPos=parPosGen(center,pNo,foldNo,ringNo,rcTmp);
for iTmp in 1:totRndNo
write(outputData,"round"*string(iTmp));
println("round"*string(iTmp));
write(outputData,"\n");
write(outputData,"Rs: "*string(Rs));
write(outputData,"\n");
write(outputData,"initial positions: ");
write(outputData,"\n");
writedlm(outputData,parPos,'\t');
write(outputData,"\n");
rcTmp=mdS(center,pNo,foldNo,ringNo,rcTmp,Rs);
parPos=parPosGen(center,pNo,foldNo,ringNo,rcTmp);
write(outputData,"equilibrium positions: ");
write(outputData,"\n");
writedlm(outputData,parPos,'\t');
write(outputData,"\n");
kMat=forceMat(parPos,Rs);
write(outputData,"force matrix: ");
write(outputData,"\n");
writedlm(outputData,kMat,'\t');
println(kMat);
write(outputData,"\n");
egv=eigen(kMat);
write(outputData,"eigenvalues: ");
write(outputData,"\n");
writedlm(outputData,round.(egv.values;digits=rndDigNo),'\t');
println(round.(egv.values,digits=rndDigNo));
write(outputData,"\n");
write(outputData,"eigenvectors: ");
write(outputData,"\n");
writedlm(outputData,round.(egv.vectors;digits=rndDigNo),'\t');
write(outputData,"\n");
write(outputData,"\n");
Rs=Rs+dRs;
flush(outputData);
end
close(outputData);
return 0;
end

```