

Example of *FP* file containing atom neighborhood fingerprints string data:

```
#
# Package = MayaChemTools 7.4
# Release Date = Oct 21, 2010
#
# TimeStamp = Fri Mar 11 14:15:27 2011
#
# FingerprintsStringType = FingerprintsVector
#
# Description = AtomNeighborhoods:AtomicInvariantsAtomTypes:MinRadiu...
# VectorStringFormat = ValuesString
# VectorValuesType = AlphaNumericalValues
#
Cmpd1 41;NR0-C.X1.BO1.H3-ATC1:NR1-C.X3.BO3.H1-ATC1:NR2-C.X1.BO1.H3-A...
Cmpd2 23;NR0-C.X1.BO1.H3-ATC1:NR1-C.X2.BO2.H2-ATC1:NR2-C.X3.BO3.H1-A...
... ..
... ..
```

Example of CSV *Text* file containing atom neighborhood fingerprints string data:

```
"CompoundID", "AtomNeighborhoodsFingerprints"
"Cmpd1", "FingerprintsVector;AtomNeighborhoods:AtomicInvariantsAtomTypes
:MinRadius0:MaxRadius2;41;AlphaNumericalValues;ValuesString;NR0-C.X1.B
O1.H3-ATC1:NR1-C.X3.BO3.H1-ATC1:NR2-C.X1.BO1.H3-ATC1:NR2-C.X3.BO4-ATC1
NR0-C.X1.BO1.H3-ATC1:NR1-C.X3.BO3.H1-ATC1:NR2-C.X1.BO1.H3-ATC1:NR2-C.X3
.BO4-ATC1 NR0-C.X2.BO2.H2-ATC1:NR1-C.X2.BO2.H2-ATC1:NR1-C.X3.BO3.H1..."
... ..
... ..
```

The current release of MayaChemTools generates the following types of atom neighborhoods fingerprints vector strings:

```
FingerprintsVector;AtomNeighborhoods:AtomicInvariantsAtomTypes:MinRadi
us0:MaxRadius2;41;AlphaNumericalValues;ValuesString;NR0-C.X1.BO1.H3-AT
C1:NR1-C.X3.BO3.H1-ATC1:NR2-C.X1.BO1.H3-ATC1:NR2-C.X3.BO4-ATC1 NR0-C.X
1.BO1.H3-ATC1:NR1-C.X3.BO3.H1-ATC1:NR2-C.X1.BO1.H3-ATC1:NR2-C.X3.BO4-A
TC1 NR0-C.X2.BO2.H2-ATC1:NR1-C.X2.BO2.H2-ATC1:NR1-C.X3.BO3.H1-ATC1:NR2
-C.X2.BO2.H2-ATC1:NR2-N.X3.BO3-ATC1:NR2-O.X1.BO1.H1-ATC1 NR0-C.X2.B...
```

```
FingerprintsVector;AtomNeighborhoods:DREIDINGAtomTypes:MinRadius0:MaxR
adius2;41;AlphaNumericalValues;ValuesString;NR0-C_2-ATC1:NR1-C_3-ATC1:
NR1-O_2-ATC1:NR1-O_3-ATC1:NR2-C_3-ATC1 NR0-C_2-ATC1:NR1-C_R-ATC1:NR1-N
_3-ATC1:NR1-O_2-ATC1:NR2-C_R-ATC3 NR0-C_3-ATC1:NR1-C_2-ATC1:NR1-C_3-AT
C1:NR2-C_3-ATC1:NR2-O_2-ATC1:NR2-O_3-ATC2 NR0-C_3-ATC1:NR1-C_3-ATC1:NR
1-N_R-ATC1:NR2-C_3-ATC1:NR2-C_R-ATC2 NR0-C_3-ATC1:NR1-C_3-ATC1:NR2-...
```

```
FingerprintsVector;AtomNeighborhoods:ESStateAtomTypes:MinRadius0:MaxRad
ius2;41;AlphaNumericalValues;ValuesString;NR0-aaCH-ATC1:NR1-aaCH-ATC1:
NR1-aasC-ATC1:NR2-aaCH-ATC1:NR2-aasC-ATC1:NR2-sF-ATC1 NR0-aaCH-ATC1:NR
1-aaCH-ATC1:NR1-aasC-ATC1:NR2-aaCH-ATC1:NR2-aasC-ATC1:NR2-sF-ATC1 NR0-
aaCH-ATC1:NR1-aaCH-ATC1:NR1-aasC-ATC1:NR2-aaCH-ATC1:NR2-aasC-ATC2 NR0-
aaCH-ATC1:NR1-aaCH-ATC1:NR1-aasC-ATC1:NR2-aaCH-ATC1:NR2-aasC-ATC2 N...
```

```
FingerprintsVector;AtomNeighborhoods:FunctionalClassAtomTypes:MinRadiu
s0:MaxRadius2;41;AlphaNumericalValues;ValuesString;NR0-Ar-ATC1:NR1-Ar-
ATC1:NR1-Ar.HBA-ATC1:NR1-None-ATC1:NR2-Ar-ATC2:NR2-None-ATC4 NR0-Ar-AT
C1:NR1-Ar-ATC2:NR1-Ar.HBA-ATC1:NR2-Ar-ATC5:NR2-None-ATC1 NR0-Ar-ATC1:N
R1-Ar-ATC2:NR1-HBD-ATC1:NR2-Ar-ATC2:NR2-None-ATC1 NR0-Ar-ATC1:NR1-Ar-A
TC2:NR1-Hal-ATC1:NR2-Ar-ATC2 NR0-Ar-ATC1:NR1-Ar-ATC2:NR1-None-ATC1:...
```

```
FingerprintsVector;AtomNeighborhoods:MMFF94AtomTypes:MinRadius0:MaxRad
ius2;41;AlphaNumericalValues;ValuesString;NR0-C5A-ATC1:NR1-C5B-ATC1:NR
1-CB-ATC1:NR1-N5-ATC1:NR2-C5A-ATC1:NR2-C5B-ATC1:NR2-CB-ATC3:NR2-CR-ATC
1 NR0-C5A-ATC1:NR1-C5B-ATC1:NR1-CR-ATC1:NR1-N5-ATC1:NR2-C5A-ATC1:NR2-C
5B-ATC1:NR2-C=ON-ATC1:NR2-CR-ATC3 NR0-C5B-ATC1:NR1-C5A-ATC1:NR1-C5B-AT
C1:NR1-C=ON-ATC1:NR2-C5A-ATC1:NR2-CB-ATC1:NR2-CR-ATC1:NR2-N5-ATC1:N...
```

```
FingerprintsVector;AtomNeighborhoods:SLogPAtomTypes:MinRadius0:MaxRadi
us2;41;AlphaNumericalValues;ValuesString;NR0-C1-ATC1:NR1-C10-ATC1:NR1-
CS-ATC1:NR2-C1-ATC1:NR2-N11-ATC1:NR2-O2-ATC1 NR0-C1-ATC1:NR1-C11-ATC1:
NR2-C1-ATC1:NR2-C21-ATC1 NR0-C1-ATC1:NR1-C11-ATC1:NR2-C1-ATC1:NR2-C21-
ATC1 NR0-C1-ATC1:NR1-C5-ATC1:NR1-CS-ATC1:NR2-C1-ATC1:NR2-O2-ATC2:NR2-O
9-ATC1 NR0-C1-ATC1:NR1-CS-ATC2:NR2-C1-ATC2:NR2-O2-ATC2 NR0-C10-ATC1...
```

```
FingerprintsVector;AtomNeighborhoods:SYBYLAtomTypes:MinRadius0:MaxRadi
```

```
us2;41;AlphaNumericalValues;ValuesString;NR0-C.2-ATC1:NR1-C.3-ATC1:NR1-O.co2-ATC2:NR2-C.3-ATC1 NR0-C.2-ATC1:NR1-C.ar-ATC1:NR1-N.am-ATC1:NR1-O.2-ATC1:NR2-C.ar-ATC3 NR0-C.3-ATC1:NR1-C.2-ATC1:NR1-C.3-ATC1:NR2-C.3-ATC1:NR2-O.3-ATC1:NR2-O.co2-ATC2 NR0-C.3-ATC1:NR1-C.3-ATC1:NR1-N.ar-ATC1:NR2-C.3-ATC1:NR2-C.ar-ATC2 NR0-C.3-ATC1:NR1-C.3-ATC1:NR2-C.3-ATC...
```

```
FingerprintsVector;AtomNeighborhoods:TPSAAtomTypes:MinRadius0:MaxRadius2;41;AlphaNumericalValues;ValuesString;NR0-N21-ATC1:NR1-None-ATC3:NR2-None-ATC5 NR0-N7-ATC1:NR1-None-ATC2:NR2-None-ATC3:NR2-O3-ATC1 NR0-None-ATC1:NR1-N21-ATC1:NR1-None-ATC1:NR2-None-ATC3 NR0-None-ATC1:NR1-N21-ATC1:NR1-None-ATC2:NR2-None-ATC6 NR0-None-ATC1:NR1-N21-ATC1:NR1-None-ATC2:NR2-None-ATC6 NR0-None-ATC1:NR1-N7-ATC1:NR1-None-ATC1:NR1-O3-AT...
```

```
FingerprintsVector;AtomNeighborhoods:UFFAtomTypes:MinRadius0:MaxRadius2;41;AlphaNumericalValues;ValuesString;NR0-C_2-ATC1:NR1-C_3-ATC1:NR1-O_2-ATC1:NR1-O_3-ATC1:NR2-C_3-ATC1 NR0-C_2-ATC1:NR1-C_R-ATC1:NR1-N_3-ATC1:NR1-O_2-ATC1:NR2-C_R-ATC3 NR0-C_3-ATC1:NR1-C_2-ATC1:NR1-C_3-ATC1:NR2-C_3-ATC1:NR2-O_2-ATC1:NR2-O_3-ATC2 NR0-C_3-ATC1:NR1-C_3-ATC1:NR1-N_R-ATC1:NR2-C_3-ATC1:NR2-C_R-ATC2 NR0-C_3-ATC1:NR1-C_3-ATC1:NR2-C_3-A...
```

OPTIONS

-a, --AtomIdentifierType *AtomicInvariantsAtomTypes | DREIDINGAtomTypes | EStateAtomTypes | FunctionalClassAtomTypes | MMFF94AtomTypes | SLogPAtomTypes | SYBYLAtomTypes | TPSAAtomTypes | UFFAtomTypes*

Specify atom identifier type to use for assignment of initial atom identifier to non-hydrogen atoms during calculation of atom neighborhoods fingerprints. Possible values in the current release are: *AtomicInvariantsAtomTypes*, *DREIDINGAtomTypes*, *EStateAtomTypes*, *FunctionalClassAtomTypes*, *MMFF94AtomTypes*, *SLogPAtomTypes*, *SYBYLAtomTypes*, *TPSAAtomTypes*, *UFFAtomTypes*. Default value: *AtomicInvariantsAtomTypes*.

--AtomicInvariantsToUse "*AtomicInvariant,AtomicInvariant...*"

This value is used during *AtomicInvariantsAtomTypes* value of a, --AtomIdentifierType option. It's a list of comma separated valid atomic invariant atom types.

Possible values for atomic invariants are: *AS, X, BO, LBO, SB, DB, TB, H, Ar, RA, FC, MN, SM*. Default value: *AS,X,BO,H,FC*.

The atomic invariants abbreviations correspond to:

AS = Atom symbol corresponding to element symbol

X<n> = Number of non-hydrogen atom neighbors or heavy atoms

BO<n> = Sum of bond orders to non-hydrogen atom neighbors or heavy atoms

LBO<n> = Largest bond order of non-hydrogen atom neighbors or heavy atoms

SB<n> = Number of single bonds to non-hydrogen atom neighbors or heavy atoms

DB<n> = Number of double bonds to non-hydrogen atom neighbors or heavy atoms

TB<n> = Number of triple bonds to non-hydrogen atom neighbors or heavy atoms

H<n> = Number of implicit and explicit hydrogens for atom

Ar = Aromatic annotation indicating whether atom is aromatic

RA = Ring atom annotation indicating whether atom is a ring

FC<+n/-n> = Formal charge assigned to atom

MN<n> = Mass number indicating isotope other than most abundant isotope

SM<n> = Spin multiplicity of atom. Possible values: 1 (singlet), 2 (doublet) or 3 (triplet)

Atom type generated by AtomTypes::AtomicInvariantsAtomTypes class corresponds to:

AS.X<n>.BO<n>.LBO<n>.<SB><n>.<DB><n>.<TB><n>.H<n>.Ar.RA.FC<+n/-n>.MN<n>.SM<n>

Except for AS which is a required atomic invariant in atom types, all other atomic invariants are optional. Atom type specification doesn't include atomic invariants with zero or undefined values.

In addition to usage of abbreviations for specifying atomic invariants, the following descriptive words are also allowed:

X : NumOfNonHydrogenAtomNeighbors or NumOfHeavyAtomNeighbors

BO : SumOfBondOrdersToNonHydrogenAtoms or SumOfBondOrdersToHeavyAtoms

LBO : LargestBondOrderToNonHydrogenAtoms or LargestBondOrderToHeavyAtoms

SB : NumOfSingleBondsToNonHydrogenAtoms or NumOfSingleBondsToHeavyAtoms

DB : NumOfDoubleBondsToNonHydrogenAtoms or NumOfDoubleBondsToHeavyAtoms

TB : NumOfTripleBondsToNonHydrogenAtoms or NumOfTripleBondsToHeavyAtoms

H : NumOfImplicitAndExplicitHydrogens

Ar : Aromatic

RA : RingAtom

FC : FormalCharge

MN : MassNumber

SM : SpinMultiplicity

AtomTypes::AtomicInvariantsAtomTypes module is used to assign atomic invariant atom types.

--FunctionalClassesToUse "*FunctionalClass1,FunctionalClass2...*"

This value is used during *FunctionalClassAtomTypes* value of a, --AtomIdentifierType option. It's a list of comma separated valid functional classes.

Possible values for atom functional classes are: *Ar, CA, H, HBA, HBD, Hal, NI, PI, RA*. Default value [Ref 24]:
HBD,HBA,PI,NI,Ar,Hal.

The functional class abbreviations correspond to:

```
HBD: HydrogenBondDonor
HBA: HydrogenBondAcceptor
PI : PositivelyIonizable
NI : NegativelyIonizable
Ar : Aromatic
Hal : Halogen
H : Hydrophobic
RA : RingAtom
CA : ChainAtom
```

Functional class atom type specification for an atom corresponds to:

```
Ar.CA.H.HBA.HBD.Hal.NI.PI.RA
```

AtomTypes::FunctionalClassAtomTypes module is used to assign functional class atom types. It uses following definitions [Ref 60-61, Ref 65-66]:

```
HydrogenBondDonor: NH, NH2, OH
HydrogenBondAcceptor: N[!H], O
PositivelyIonizable: +, NH2
NegativelyIonizable: -, C(=O)OH, S(=O)OH, P(=O)OH
```

--CompoundID *DataFieldName* or *LabelPrefixString*

This value is --CompoundIDMode specific and indicates how compound ID is generated.

For *DataField* value of --CompoundIDMode option, it corresponds to datafield label name whose value is used as compound ID; otherwise, it's a prefix string used for generating compound IDs like *LabelPrefixString<Number>*. Default value, *Cmpd*, generates compound IDs which look like *Cmpd<Number>*.

Examples for *DataField* value of --CompoundIDMode:

```
MolID
ExtReg
```

Examples for *LabelPrefix* or *MolNameOrLabelPrefix* value of --CompoundIDMode:

```
Compound
```

The value specified above generates compound IDs which correspond to *Compound<Number>* instead of default value of *Cmpd<Number>*.

--CompoundIDLabel *text*

Specify compound ID column label for FP or CSV/TSV text file(s) used during *CompoundID* value of --DataFieldsMode option. Default: *CompoundID*.

--CompoundIDMode *DataField | MolName | LabelPrefix | MolNameOrLabelPrefix*

Specify how to generate compound IDs and write to FP or CSV/TSV text file(s) along with generated fingerprints for *FP | text | all* values of --output option: use a *SDFFile(s)* datafield value; use molname line from *SDFFile(s)*; generate a sequential ID with specific prefix; use combination of both *MolName* and *LabelPrefix* with usage of *LabelPrefix* values for empty molname lines.

Possible values: *DataField | MolName | LabelPrefix | MolNameOrLabelPrefix*. Default: *LabelPrefix*.

For *MolNameAndLabelPrefix* value of --CompoundIDMode, molname line in *SDFFile(s)* takes precedence over sequential compound IDs generated using *LabelPrefix* and only empty molname values are replaced with sequential compound IDs.

This is only used for *CompoundID* value of --DataFieldsMode option.

--DataFields "*FieldLabel1,FieldLabel2,...*"

Comma delimited list of *SDFFile(s)* data fields to extract and write to CSV/TSV text file(s) along with generated fingerprints for *text | all* values of --output option.

This is only used for *Specify* value of --DataFieldsMode option.

Examples:

```
Extreg
MolID,CompoundName
```

-d, --DataFieldsMode *All | Common | Specify | CompoundID*

Specify how data fields in *SDFFile(s)* are transferred to output CSV/TSV text file(s) along with generated fingerprints for *text | all* values of --output option: transfer all SD data field; transfer SD data files common to all compounds; extract specified data fields; generate a compound ID using molname line, a compound prefix, or a combination of both.

Possible values: *All | Common | specify | CompoundID*. Default value: *CompoundID*.

-f, --Filter *Yes | No*

Specify whether to check and filter compound data in *SDFFile(s)*. Possible values: *Yes* or *No*. Default value: *Yes*.

By default, compound data is checked before calculating fingerprints and compounds containing atom data

corresponding to non-element symbols or no atom data are ignored.

--FingerprintsLabel *text*

SD data label or text file column label to use for fingerprints string in output SD or CSV/TSV text file(s) specified by --output. Default value: *AtomNeighborhoodsFingerprints*.

-h, --help

Print this help message.

-k, --KeepLargestComponent *Yes / No*

Generate fingerprints for only the largest component in molecule. Possible values: *Yes or No*. Default value: *Yes*.

For molecules containing multiple connected components, fingerprints can be generated in two different ways: use all connected components or just the largest connected component. By default, all atoms except for the largest connected component are deleted before generation of fingerprints.

--MinNeighborhoodRadius *number*

Minimum atom neighborhood radius for generating atom neighborhoods. Default value: *0*. Valid values: positive integers and less than --MaxNeighborhoodRadius. Neighborhood radius of zero corresponds to list of non-hydrogen atoms.

--MaxNeighborhoodRadius *number*

Maximum atom neighborhood radius for generating atom neighborhoods. Default value: *2*. Valid values: positive integers and greater than --MinNeighborhoodRadius.

--OutDelim *comma | tab | semicolon*

Delimiter for output CSV/TSV text file(s). Possible values: *comma, tab, or semicolon* Default value: *comma*.

--output *SD | FP | text | all*

Type of output files to generate. Possible values: *SD, FP, text, or all*. Default value: *text*.

-o, --overwrite

Overwrite existing files.

-q, --quote *Yes / No*

Put quote around column values in output CSV/TSV text file(s). Possible values: *Yes or No*. Default value: *Yes*.

-r, --root *RootName*

New file name is generated using the root: <Root>.<Ext>. Default for new file names: <SDFFileName><AtomNeighborhoodsFP>.<Ext>. The file type determines <Ext> value. The sdf, fpf, csv, and tsv <Ext> values are used for SD, comma/semicolon, and tab delimited text files, respectively. This option is ignored for multiple input files.

-w, --WorkingDir *DirName*

Location of working directory. Default: current directory.

EXAMPLES

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using atomic invariants atom types in vector string format and create a SampleANFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -r SampleANFP -o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using DREIDING atom types in vector string format and create a SampleANFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a DREIDINGAtomTypes -r SampleANFP
-o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using EStateAtomTypes types in vector string format and create a SampleANFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a EStateAtomTypes -r SampleANFP
-o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using SYBYL atom types in vector string format and create a SampleANFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a SYBYLAtomTypes -r SampleANFP
-o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using FunctionalClass atom

types in vector string format and create a SampleANFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a FunctionalClassAtomTypes
-r SampleANFP -o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using MMFF94 atom types in vector string format and create a SampleANFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a MMFF94AtomTypes -r SampleANFP
-o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using SLogP atom types in vector string format and create a SampleANFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a SLogPAtomTypes -r SampleANFP
-o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using SYBYL atom types in vector string format and create a SampleANFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a SYBYLAtomTypes -r SampleANFP
-o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using TPSA atom types in vector string format and create a SampleANFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a TPSAAtomTypes -r SampleANFP
-o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using UFF atom types in vector string format and create a SampleANFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a UFFAtomTypes -r SampleANFP
-o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using atomic invariants atom types in vector string format and create SampleANFP.sdf, SampleANFP.pf and SampleANFP.csv files containing sequential compound IDs in CSV file along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl --output all -r SampleANFP
-o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 1 to 3 using atomic invariants atom types in vector string format and create a SampleANFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a AtomicInvariantsAtomTypes
--MinNeighborhoodRadius 1 --MaxNeighborhoodRadius 3 -r SampleANFP
-o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using only AS,X atomic invariants atom types in vector string format and create a SampleANFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a AtomicInvariantsAtomTypes
--AtomicInvariantsToUse "AS,X" --MinNeighborhoodRadius 0
--MaxNeighborhoodRadius 3 -r SampleANFP -o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using atomic invariants atom types in vector string format and create a SampleANFP.csv file containing compound ID from molecule name line along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a AtomicInvariantsAtomTypes
--DataFieldsMode CompoundID --CompoundIDMode MolName
-r SampleANFP -o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using atomic invariants atom types in vector string format and create a SampleANFP.csv file containing compound IDs using specified data field along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a AtomicInvariantsAtomTypes
--DataFieldsMode CompoundID --CompoundIDMode DataField --CompoundID
Mol_ID -r SampleANFP -o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using atomic invariants atom types in vector string format and create a SampleANFP.csv file containing compound ID using combination of molecule name line and an explicit compound prefix along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a AtomicInvariantsAtomTypes
--DataFieldsMode CompoundID --CompoundIDMode MolnameOrLabelPrefix
--CompoundID Cmpd --CompoundIDLabel MolID -r SampleANFP -o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using atomic invariants atom types in vector string format and create a SampleANFP.csv file containing specific data fields columns along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a AtomicInvariantsAtomTypes
--DataFieldsMode Specify --DataFields Mol_ID -r SampleANFP
-o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using atomic invariants atom types in vector string format and create a SampleANFP.csv file containing common data fields columns along with fingerprints vector strings data, type:

```
% AtomNeighborhoodsFingerprints.pl -a AtomicInvariantsAtomTypes
--DataFieldsMode Common -r SampleANFP -o Sample.sdf
```

To generate atom neighborhoods fingerprints corresponding to atom neighborhood radii from 0 to 2 using atomic invariants atom types in vector string format and create SampleANFP.sdf, SampleANFP.fpf and SampleANFP.csv files containing all data fields columns in CSV file along with fingerprints data, type:

```
% AtomNeighborhoodsFingerprints.pl -a AtomicInvariantsAtomTypes
--DataFieldsMode All --output all -r SampleANFP
-o Sample.sdf
```

AUTHOR

Manish Sud <msud@san.rr.com>

SEE ALSO

InfoFingerprintsFiles.pl, SimilarityMatricesFingerprints.pl, SimilaritySearchingFingerprints.pl, ExtendedConnectivityFingerprints.pl, MACCSKeysFingerprints.pl, PathLengthFingerprints.pl, TopologicalAtomPairsFingerprints.pl, TopologicalAtomTorsionsFingerprints.pl, TopologicalPharmacophoreAtomPairsFingerprints.pl, TopologicalPharmacophoreAtomTripletsFingerprints.pl

COPYRIGHT

Copyright (C) 2004-2012 Manish Sud. All rights reserved.

This file is part of MayaChemTools.

MayaChemTools is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version.