NAME

SequenceFileUtil

SYNOPSIS

use SequenceFileUtil ;

use SequenceFileUtil qw(:all);

DESCRIPTION

SequenceFileUtil module provides the following functions:

AreSequenceLengthsIdentical, CalcuatePercentSequenceIdentity, CalculatePercentSequenceIdentityMatrix, GetLongestSequence, GetSequenceLength, GetShortestSequence, IsClustalWSequenceFile, IsGapResidue, IsMSFSequenceFile, IsPIRFastaSequenceFile, IsPearsonFastaSequenceFile, IsSupportedSequenceFile, ReadClustalWSequenceFile, ReadMSFSequenceFile, ReadPIRFastaSequenceFile, ReadPearsonFastaSequenceFile, ReadSequenceFile, RemoveSequenceAlignmentGapColumns, RemoveSequenceGaps, WritePearsonFastaSequenceFile SequenceFileUtil module provides various methods to process sequence files and retreive appropriate information.

FUNCTIONS

AreSequenceLengthsIdentical

        $Status = AreSequenceLengthsIdentical($SequencesDataRef);

Checks the lengths of all the sequences available in *SequencesDataRef* and returns 1 or 0 based whether lengths of all the sequence is same.

CalcuatePercentSequenceIdentity

        $PercentIdentity =
          AreSequenceLengthsIdenticalAreSequenceLengthsIdentical(
            $Sequence1, $Sequence2, [$IgnoreGaps, $Precision]);

Returns percent identity between *Sequence1* and *Sequence2*. Optional arguments *IgnoreGaps* and *Precision* control handling of gaps in sequences and precision of the returned value. By default, gaps are ignored and precision is set up to 1 decimal.

CalculatePercentSequenceIdentityMatrix

        $IdentityMatrixDataRef = CalculatePercentSequenceIdentityMatrix(
                                    $SequencesDataRef, [$IgnoreGaps,
                                    $Precision]);

Calculate pairwise percent identity between all the sequences available in *SequencesDataRef* and returns a reference to identity matrix hash. Optional arguments *IgnoreGaps* and *Precision* control handling of gaps in sequences and precision of the returned value. By default, gaps are ignored and precision is set up to 1 decimal.

GetSequenceLength

        $SeqquenceLength = GetSequenceLength($Sequence, [$IgnoreGaps]);

Returns length of the specified sequence. Optional argument *IgnoreGaps* controls handling of gaps. By default, gaps are ignored.

GetShortestSequence

        ($ID, $Sequence, $SeqLen, $Description) = GetShortestSequence(
            $SequencesDataRef, [$IgnoreGaps]);

Checks the lengths of all the sequences available in $SequencesDataRef and returns $ID, $Sequence, $SeqLen, and $Description values for the shortest sequence. Optional arguments $IgnoreGaps controls handling of gaps in sequences. By default, gaps are ignored.

GetLongestSequence

        ($ID, $Sequence, $SeqLen, $Description) = GetLongestSequence(
            $SequencesDataRef, [$IgnoreGaps]);

Checks the lengths of all the sequences available in *SequencesDataRef* and returns ID, Sequence, SeqLen, and Description values for the longest sequence. Optional argument $*IgnoreGaps* controls handling of gaps in sequences. By default, gaps are ignored.

IsGapResidue

> ```
> $Status = AreSequenceLengthsIdentical($Residue);
> ```

Returns 1 or 0 based on whether *Residue* corresponds to a gap. Any character other than A to Z is considered a gap residue.

IsSupportedSequenceFile

> ```
> $Status = IsSupportedSequenceFile($SequenceFile);
> ```

Returns 1 or 0 based on whether *SequenceFile* corresponds to a supported sequence format.

IsClustalWSequenceFile

> ```
> $Status = IsClustalWSequenceFile($SequenceFile);
> ```

Returns 1 or 0 based on whether *SequenceFile* corresponds to Clustal sequence alignment format.

IsPearsonFastaSequenceFile

> ```
> $Status = IsPearsonFastaSequenceFile($SequenceFile);
> ```

Returns 1 or 0 based on whether *SequenceFile* corresponds to Pearson FASTA sequence format.

IsPIRFastaSequenceFile

> ```
> $Status = IsPIRFastaSequenceFile($SequenceFile);
> ```

Returns 1 or 0 based on whether *SequenceFile* corresponds to PIR FASTA sequence format.

IsMSFSequenceFile

> ```
> $Status = IsClustalWSequenceFile($SequenceFile);
> ```

Returns 1 or 0 based on whether *SequenceFile* corresponds to MSF sequence alignment format.

ReadSequenceFile

> ```
> $SequenceDataMapRef = ReadSequenceFile($SequenceFile);
> ```

Reads *SequenceFile* and returns reference to a hash containing following key/value pairs:

> ```
> $SequenceDataMapRef->{IDs} - Array of sequence IDs
> $SequenceDataMapRef->{Count} - Number of sequences
> $SequenceDataMapRef->{Description}{$ID} - Sequence description
> $SequenceDataMapRef->{Sequence}{$ID} - Sequence for a specific ID
> $SequenceDataMapRef->{Sequence}{InputFileType} - File format
> ```

ReadClustalWSequenceFile

> ```
> $SequenceDataMapRef = ReadClustalWSequenceFile($SequenceFile);
> ```

Reads ClustalW *SequenceFile* and returns reference to a hash containing following key/value pairs as describes in ReadSequenceFile method.

ReadMSFSequenceFile

> ```
> $SequenceDataMapRef = ReadMSFSequenceFile($SequenceFile);
> ```

Reads MSF *SequenceFile* and returns reference to a hash containing following key/value pairs as describes in ReadSequenceFile method.

ReadPIRFastaSequenceFile

> ```
> $SequenceDataMapRef = ReadPIRFastaSequenceFile($SequenceFile);
> ```

Reads PIR FASTA *SequenceFile* and returns reference to a hash containing following key/value pairs as describes in ReadSequenceFile method.

ReadPearsonFastaSequenceFile

> ```
> $SequenceDataMapRef = ReadPearsonFastaSequenceFile($SequenceFile);
> ```

Reads Pearson FASTA *SequenceFile* and returns reference to a hash containing following key/value pairs as describes in ReadSequenceFile method.

RemoveSequenceGaps

```
$SeqWithoutGaps = RemoveSequenceGaps($Sequence);
```

Removes gaps from *Sequence* and return a sequence without any gaps.

RemoveSequenceAlignmentGapColumns

```
$NewAlignmentDataMapRef = RemoveSequenceAlignmentGapColumns(
                          $AlignmentDataMapRef);
```

Using input alignment data map ref containing following keys, generate a new hash with same set of keys after residue columns containing only gaps have been removed:

```
{IDs} : Array of IDs in order as they appear in file
{Count}: ID count
{Description}{$ID} : Description data
{Sequence}{$ID} : Sequence data
```

WritePearsonFastaSequenceFile

```
WritePearsonFastaSequenceFile($SequenceFileName, $SequenceDataRef,
                              [$MaxLength]);
```

Using sequence data specified via *SequenceDataRef*, write out a Pearson FASTA sequence file. Optional argument *MaxLength* controls maximum length sequence in each line; default is 80.

AUTHOR

Manish Sud <msud@san.rr.com>

SEE ALSO

PDBFileUtil.pm

COPYRIGHT