Projekt_peptydy

Wygenerowano przez Doxygen 1.8.13

Spis treści

1 Indeks przestrzeni nazw

1.1 Pakiety

Oto lista pakietów wraz z krótkim opisem (o ile jest dostępny):

2 Indeks klas

2.1 Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

3 Indeks plików

3.1 Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

4 Dokumentacja przestrzeni nazw

4.1 Dokumentacja przestrzeni nazw program

Komponenty

- class **atom**

  Atomy.
- class **monomer**

  Monomery.
- class **peptide**

  Peptydy.
- class **vector**

  Wektory w układzie kartezjańskim.

Funkcje

- def **transpose** (m)

  Transpozyjcja macierzy.
- def **dihedral_angle** (b1, b2, b3)

  Wyznaczanie kąta torsyjnego.
- def **upload_data** (**table**, **l1**, **l2**, **l3**, **l4**, **l5**, **l7**, **l8**)
- def **make_output** (peptyd, **i**, **j**, **n**)

Zmienne

- list **table** = [ ]
- int **number_of_lines** = 0
- dictionary **monomers_list** = {}
- list **l1** = [ ]
- list **l2** = [ ]
- list **l3** = [ ]
- list **l4** = [ ]
- list **l5** = [ ]
- int **l6** = 0
- list **l7** = [ ]
- list **l8** = [ ]
- **words** = re.split("\s+",line.strip())
- def **monomer_i** = **upload_data**(**table**, **l1**, **l2**, **l3**, **l4**, **l5**, **l7**, **l8**)
- **data** = open(sys.argv[1])
- string **words2** = ""
- list **table2** = [ ]
- **aa** = **peptide**()
- **f** = open('output.pdb', 'w')
- int **i** = 0
- int **j** = 0
- int **n** = 0

4.1.1 Dokumentacja funkcji

4.1.1.1 dihedral_angle()

```
def program.dihedral_angle (
              b1,
              b2,
              b3 )
```

Wyznaczanie kąta torsyjnego.

Parametry

b1,b2,b3	trzy wektory pomiędzy, którymi liczony jest kąt torsyjny

Zwraca

wartość kąta torsyjnego

Autor

Bernadeta Nowosielska

4.1.1.2 make_output()

```
def program.make_output (
              peptyd,
              i,
              j,
              n )
```

4.1.1.3 transpose()

```
def program.transpose (
              m )
```

Transpozyjcja macierzy.

Parametry

m	maciwerz, która ma zostać zmieniona

Zwraca

macierz transponowana

Autor

Bernadeta Nowosielska

4.1.1.4 upload_data()

```
def program.upload_data (
              table,
              l1,
              l2,
              l3,
              l4,
              l5,
              l7,
              l8 )
```

4.1.2 Dokumentacja zmiennych

4.1.2.1 aa

```
program.aa =  peptide()
```

4.1.2.2 data

```
program.data = open(sys.argv[1])
```

4.1.2.3 f

```
program.f = open('output.pdb', 'w')
```

4.1.2.4 i

```
int program.i = 0
```

4.1.2.5 j

```
int program.j = 0
```

4.1.2.6 l1

```
list program.l1 = []
```

4.1.2.7 l2

```
list program.l2 = []
```

4.1.2.8 l3

```
list program.l3 = []
```

4.1.2.9 l4

```
list program.l4 = []
```

4.1.2.10 l5

```
list program.l5 = []
```

4.1.2.11 l6

```
int program.l6 = 0
```

4.1.2.12 l7

```
list program.l7 = []
```

4.1.2.13 l8

```
list program.l8 = []
```

4.1.2.14 monomer_i

```
def program.monomer_i =  upload_data( table,  l1,  l2,  l3,  l4,  l5,  l7,  l8)
```

4.1.2.15 monomers_list

```
dictionary program.monomers_list = {}
```

4.1.2.16 n

```
int program.n = 0
```

4.1.2.17 number_of_lines

```
int program.number_of_lines = 0
```

4.1.2.18 table

```
list program.table = []
```

4.1.2.19 table2

```
list program.table2 = []
```

4.1.2.20 words

```
program.words = re.split("\s+",line.strip())
```

4.1.2.21 words2

```
program.words2 = ""
```

5 Dokumentacja klas

5.1 Dokumentacja klasy program.atom

Atomy.

Metody publiczne

- def **__init__** (__self__, a, b, c, **ID**, **element**, **znacznik**)
- def **scalar_prod** (__self__, p)
- def **rotate** (__self__, m)
- def **translation** (__self__, v)
- def **vector_prod** (__self__, p)
- def **__str__** (__self__)

Statyczne atrybuty publiczne

- int **x** = 0
- int **y** = 0
- int **z** = 0
- string **ID** = ""
- string **element** = ""
- string **znacznik** = ""

5.1.1 Opis szczegółowy

Atomy.

Autor

Bernadeta Nowosielska

Parametry

x,y,z	współrzędne w układzie kartezjanskim
ID	unikalny identyfikator
element	pierwiastek chemiczny
znacznik	atomy "funkcyjne"

5.1.2 Dokumentacja konstruktora i destruktora

5.1.2.1 __init__()

```
def program.atom.__init__ (
              __self__,
              a,
              b,
              c,
              ID,
              element,
              znacznik )
```

5.1.3 Dokumentacja funkcji składowych

5.1.3.1 __str__()

```
def program.atom.__str__ (
              __self__ )
```

5.1.3.2 rotate()

```
def program.atom.rotate (
              __self__,
              m )
```

5.1.3.3 scalar_prod()

```
def program.atom.scalar_prod (
              __self__,
              p )
```

5.1.3.4 translation()

```
def program.atom.translation (
              __self__,
              v )
```

5.1.3.5 vector_prod()

```
def program.atom.vector_prod (
              __self__,
              p )
```

5.1.4 Dokumentacja atrybutów składowych

5.1.4.1 element

```
string program.atom.element = ""  [static]
```

5.1.4.2 ID

```
string program.atom.ID = ""  [static]
```

5.1.4.3 x

```
int program.atom.x = 0  [static]
```

5.1.4.4 y

```
int program.atom.y = 0  [static]
```

5.1.4.5 z

```
int program.atom.z = 0  [static]
```

5.1.4.6 znacznik

```
string program.atom.znacznik = ""  [static]
```

Dokumentacja dla tej klasy została wygenerowana z pliku:

- **program.py**

5.2 Dokumentacja klasy program.monomer

Monomery.

Metody publiczne

- def **__init__** (__self__, **ID**, noa, atoms, nob, bonds, wazne_atomy)
  Konstuktor.
- def **replicate** (__self__)
  Tworzy wierną kopję monomeru.
- def **rotate_all** (__self__, m)
  Obrót wszystkich atomu monomeru.
- def **translation_all** (__self__, v)
  Przesunięcie wszystkich atomów monomeru.
- def **system_C** (__self__)
  Tworzy macierz układu współrzędnyc na C.
- def **system_N** (__self__)
  Tworzy macierz układu współrzędnyc na C.
- def **remove** (__self__, **ID**)
  Usuwa atom.
- def **__str__** (__self__)
  Pozwala na wydrukowanie monomru.

Statyczne atrybuty publiczne

- string **ID** = ""
- int **number_of_atoms** = 0
- int **number_of_bonds** = 0

5.2.1 Opis szczegółowy

Monomery.

Autor

Bernadeta Nowosielska

Parametry

ID	nazwa monomeru
number_of_atoms	liczba atomów w monomerze
number_of_bonds	liczba wiązań w monomerze

5.2.2 Dokumentacja konstruktora i destruktora

5.2.2.1 __init__()

```
def program.monomer.__init__ (
              __self__,
              ID,
              noa,
              atoms,
              nob,
              bonds,
              wazne_atomy )
```

Konstuktor.

Parametry

ID	nazwa
noa	liczba atomów w monomerze
atoms	tablica zawierająca obiekty klasy atom
nob	liczba wiązań w monomerze
bonds	tablica wiązań
wazne_atomy	tablica atomów funkcyjnych

5.2.3 Dokumentacja funkcji składowych

5.2.3.1 __str__()

```
def program.monomer.__str__ (
              __self__ )
```

Pozwala na wydrukowanie monomru.

5.2.3.2 remove()

```
def program.monomer.remove (
              __self__,
              ID )
```

Usuwa atom.

Parametry

ID	ID atomu który będzie usunięty

5.2.3.3 replicate()

```
def program.monomer.replicate (
              __self__ )
```

Tworzy wierną kopję monomeru.

5.2.3.4 rotate_all()

```
def program.monomer.rotate_all (
              __self__,
              m )
```

Obrót wszystkich atomu monomeru.

Parametry

m	macierz obrotu

5.2.3.5 system_C()

```
def program.monomer.system_C (
              __self__ )
```

Tworzy macierz układu współrzędnyc na C.

Zwraca

Macierz układu

Tworzy macierz układu współrzędnych na karbonylowym atomie węgla. Tak aby oś x znajdowała się na wiązaniu

5.2.3.6 system_N()

```
def program.monomer.system_N (
              __self__ )
```

Tworzy macierz układu współrzędnyc na C.

Zwraca

Macierz układu

Tworzy macierz układu współrzędnych na karbonylowym atomie węgla. Tak aby oś x znajdowała się na wiązaniu

5.2.3.7 translation_all()

```
def program.monomer.translation_all (
             __self__,
             v )
```

Przesunięcie wszystkich atomów monomeru.

Parametry

v	wektor, o który następuje przesunięcie

5.2.4 Dokumentacja atrybutów składowych

5.2.4.1 ID

```
string program.monomer.ID = ""  [static]
```

5.2.4.2 number_of_atoms

```
int program.monomer.number_of_atoms = 0  [static]
```

5.2.4.3 number_of_bonds

```
int program.monomer.number_of_bonds = 0  [static]
```

Dokumentacja dla tej klasy została wygenerowana z pliku:

- **program.py**

5.3 Dokumentacja klasy program.peptide

Peptydy.

Metody publiczne

- def **start** (__self__, monomer1)
  Dodaje pierwszy monomer do peptydu.
- def **add** (__self__, monomer3, omega_i, fi_i, psi_i)
  Dodaje kolejne monomery.
- def **__str__** (__self__)
  Pozwala na wydrukowanie peptydu.
- def **rotate_da** (__self__, type_, angle, N, C, m3)
  Obraca o kąt torsyjny.

Statyczne atrybuty publiczne

- list **coord** = [ ]
- list **A** = [ ]

5.3.1 Opis szczegółowy

Peptydy.

Autor

Bernadeta Nowosielska

Parametry

coord	tablica zawierajaca tablicę, która zawiera ID monomeru, jego licznik oraz tablię obiektów klasy atom.

5.3.2 Dokumentacja funkcji składowych

5.3.2.1 __str__()

```
def program.peptide.__str__ (
              __self__ )
```

Pozwala na wydrukowanie peptydu.

5.3.2.2 add()

```
def program.peptide.add (
              __self__,
              monomer3,
              omega_i,
              fi_i,
              psi_i )
```

Dodaje kolejne monomery.

Parametry

monomer3	dodawany monomer
omega_i,fi_↩ i,psi_i	kąty torsyjne, które zostaną ustawione

5.3.2.3 rotate_da()

```
def program.peptide.rotate_da (
              __self__,
              type_,
              angle,
              N,
              C,
              m3 )
```

Obraca o kąt torsyjny.

Parametry

*type_*	wybrany kąt torsyjn "omega", "fi" lub "psi"
angle	wartość o którą nastąpi obrót
N,C	końcowy i początkowy atom wiązania, na którym będzie następować obrót (obiekt klasy atom)
m3	monomer, który będzie obracany

5.3.2.4 start()

```
def program.peptide.start (
              __self__,
              monomer1 )
```

Dodaje pierwszy monomer do peptydu.

Parametry

monomer1	obiekt klasy monomer

5.3.3 Dokumentacja atrybutów składowych

5.3.3.1 A

```
list program.peptide.A = []  [static]
```

5.3.3.2 coord

```
list program.peptide.coord = []  [static]
```

Dokumentacja dla tej klasy została wygenerowana z pliku:

- **program.py**

5.4 Dokumentacja klasy program.vector

Wektory w układzie kartezjańskim.

Metody publiczne

- def **__init__** (__self__, a, b, c)
  Konstuktor.
- def **leng** (__self__)
  Długość wektora.
- def **vector_prod** (__self__, p)
  Iloczyn wektorowy.
- def **scalar_prod** (__self__, p)
  Iloczyn skalarny.
- def **__str__** (__self__)
  Pozwala na wydrukowanie wektora.

Statyczne atrybuty publiczne

- int **x** = 0
- int **y** = 0
- int **z** = 0

5.4.1 Opis szczegółowy

Wektory w układzie kartezjańskim.

Autor

Bernadeta Nowosielska

Parametry

x,y,z	współrzędne w układzie karteziańskim

5.4.2 Dokumentacja konstruktora i destruktora

5.4.2.1 __init__()

```
def program.vector.__init__ (
            __self__,
            a,
            b,
            c )
```

Konstuktor.

Parametry

a,b,c	pobierane współrzędne

5.4.3 Dokumentacja funkcji składowych

5.4.3.1 __str__()

```
def program.vector.__str__ (
              __self__ )
```

Pozwala na wydrukowanie wektora.

5.4.3.2 leng()

```
def program.vector.leng (
              __self__ )
```

Długość wektora.

Zwraca

długość wekotora

5.4.3.3 scalar_prod()

```
def program.vector.scalar_prod (
              __self__,
              p )
```

Iloczyn skalarny.

Parametry

p	atom lub wektor

Zwraca

wynik iloczynu skalarnego

5.4.3.4 vector_prod()

```
def program.vector.vector_prod (
              __self__,
              p )
```

Iloczyn wektorowy.

Parametry

p	atom lub wektor

Zwraca

obiekt klasy wektor

5.4.4 Dokumentacja atrybutów składowych

5.4.4.1 x

```
int program.vector.x = 0  [static]
```

5.4.4.2 y

```
int program.vector.y = 0  [static]
```

5.4.4.3 z

```
int program.vector.z = 0  [static]
```

Dokumentacja dla tej klasy została wygenerowana z pliku:

- **program.py**

6 Dokumentacja plików

6.1 Dokumentacja pliku program.py

Komponenty

- class **program.vector**
  Wektory w układzie kartezjańskim.
- class **program.atom**
  Atomy.
- class **program.monomer**
  Monomery.
- class **program.peptide**
  Peptydy.

Przestrzenie nazw

- **program**

Funkcje

- def **program.transpose** (m)

  Transpozyjcja macierzy.
- def **program.dihedral_angle** (b1, b2, b3)

  Wyznaczanie kąta torsyjnego.
- def **program.upload_data** (table, l1, l2, l3, l4, l5, l7, l8)
- def **program.make_output** (peptyd, i, j, n)

Zmienne

- list **program.table** = [ ]
- int **program.number_of_lines** = 0
- dictionary **program.monomers_list** = {}
- list **program.l1** = [ ]
- list **program.l2** = [ ]
- list **program.l3** = [ ]
- list **program.l4** = [ ]
- list **program.l5** = [ ]
- int **program.l6** = 0
- list **program.l7** = [ ]
- list **program.l8** = [ ]
- **program.words** = re.split("\s+",line.strip())
- def **program.monomer_i** = upload_data(table, l1, l2, l3, l4, l5, l7, l8)
- **program.data** = open(sys.argv[1])
- string **program.words2** = ""
- list **program.table2** = [ ]
- **program.aa** = peptide()
- **program.f** = open('output.pdb', 'w')
- int **program.i** = 0
- int **program.j** = 0
- int **program.n** = 0