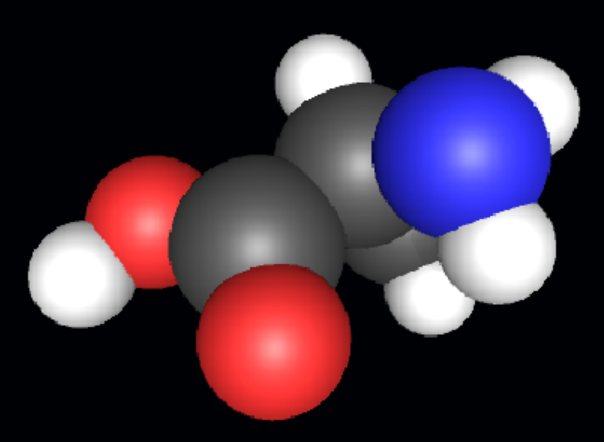
**Практика 3 февраля 2021, РГПУ**

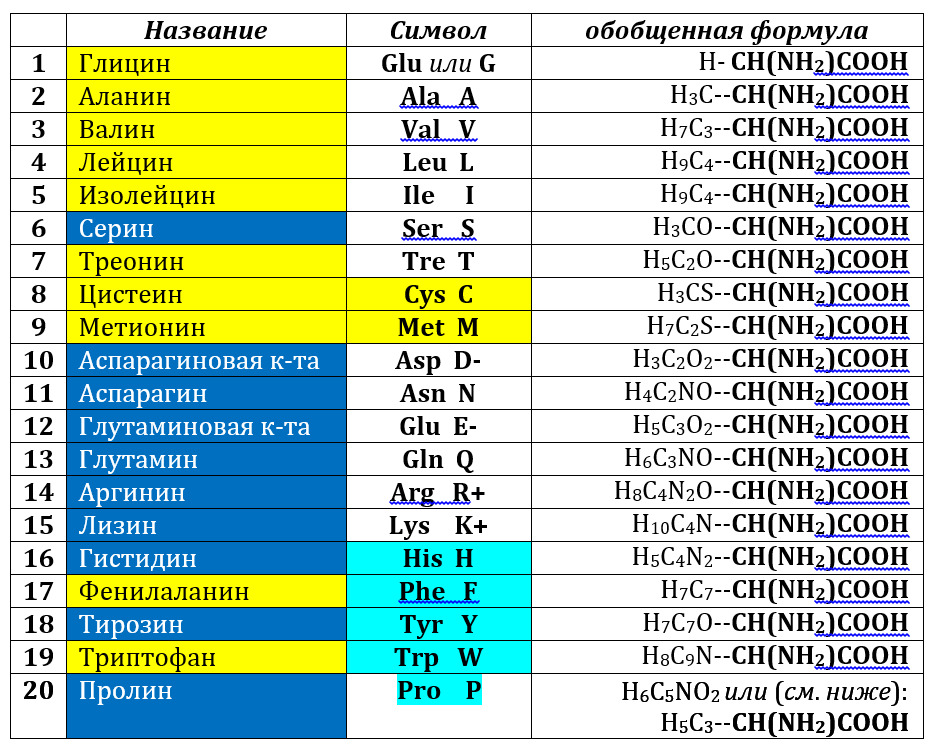
**Задача: Сделать программу для 3D визуализации молекул основных аминокислот.   
Результатом является программа в среде Windows.**

Молекула выбирается из списка на экране и помещается в центр поля.

Внешний вид – шарики соответствующих цветов и размеров:



|  |  |  |  |
| --- | --- | --- | --- |
| Элемент | Символ | Цвет | Радиус |
| Водород | H | White | 0.46 |
| Углерод | C | Gray | 0.76 |
| Азот | N | Blue | 0.71 |
| Кислород | O | Red | 0.60 |

* Ввести коэффициент пропорциональности для видимого размера атомов, чтобы их можно было менять, используя клавиши LeftCtrl + ArrowUp и LeftCtrl + ArrowDown
* Использовать Scroll мышки для приближения/удаления молекулы
* Использовать движение мышки с удержанием левой кнопки для вращения молекулы вокруг вертикальной и горизонтальной осей.
* Для разработки приложения использовать Unity 5 с встроенным скритовым языком .Net/C#.
* Набор молекул предоставляется в виде текстовых файлов формата .PDB:   
  

ЭТАПЫ

1. Установить на компьютер систему Unity 5. Пример как это сделать см.  
   <https://youtu.be/p7vajDB_dig>  
   Рекомендую для начала сделать кубик и научиться им манипулировать с использованием C#.  
   При создании скриптового приложения генерируются два файла   
   Start() и Update().  
   В модуль **Start()** заносится все разовое, инициализация, чтение файла и т.д.  
   Модуль **Update()** используется для покадрового отображения сцены. К примеру, для поворотов молекулы. Изменили угол поворота - изображение тут же изменилось.
2. Считать выбранный файл с молекулой построчно и записать его в объект класса CMolecula.   
   Вид файла:

HEADER GLY

REMARK 1

REMARK 1 -ISIS- 3D

REMARK 1

REMARK 1 10 9 0 0 0 0 0 0 0 0 0

REMARK 2

REMARK 2 BOND 1 2 1 0 0 0 1.47

REMARK 2 BOND 1 6 1 0 0 0 1.008

REMARK 2 BOND 1 7 1 0 0 0 1.009

REMARK 2 BOND 2 3 1 0 0 0 1.507

REMARK 2 BOND 2 8 1 0 0 0 1.09

REMARK 2 BOND 2 9 1 0 0 0 1.09

REMARK 2 BOND 3 4 2 0 0 0 1.208

REMARK 2 BOND 3 5 1 0 0 0 1.342

REMARK 2 BOND 5 10 1 0 0 0 0.9673

REMARK 3

REMARK 3 ANGLE 111.02 1 2 6

REMARK 3 ANGLE 111.04 1 6 7

REMARK 3 ANGLE 110.94 1 7 2

REMARK 3 ANGLE 109.44 2 1 3

REMARK 3 ANGLE 109.47 2 3 8

REMARK 3 ANGLE 109.48 2 8 1

REMARK 3 ANGLE 109.45 2 1 9

REMARK 3 ANGLE 109.47 2 3 9

REMARK 3 ANGLE 109.51 2 8 9

REMARK 3 ANGLE 120.05 3 2 4

REMARK 3 ANGLE 119.98 3 4 5

REMARK 3 ANGLE 119.97 3 5 2

REMARK 3 ANGLE 117.04 5 3 10

ATOM 1 N GLY A 1 1.931 0.09 -0.034 1.00 0.00 N

ATOM 2 CA GLY A 1 0.761 -0.799 -0.008 1.00 0.00 C

ATOM 3 C GLY A 1 -0.498 0.029 -0.005 1.00 0.00 C

ATOM 4 O GLY A 1 -0.429 1.235 -0.023 1.00 0.00 O

ATOM 5 OXT GLY A 1 -1.697 -0.574 0.018 1.00 0.00 O

ATOM 6 H GLY A 1 1.91 0.738 0.738 1.00 0.00 H

ATOM 7 H2 GLY A 1 2.788 -0.442 -0.037 1.00 0.00 H

ATOM 8 HA2 GLY A 1 0.772 -1.44 -0.889 1.00 0.00 H

ATOM 9 HA3 GLY A 1 0.793 -1.415 0.891 1.00 0.00 H

ATOM 10 HXT GLY A 1 -2.477 -0.002 0.019 1.00 0.00 H

TER 11 GLY A 1

END

Все строки, которые не начинаются на «ATOM», игнорируются. В данном случае, мы видим, что молекула содержит всего 10 атомов. Все атомы в аминокислотах имеют уникальные имена – N,CA,C,O,OXT… . А, поскольку аминокислоты имеют свойство образовывать цепочки, то указывается имя в цепочке (Здесь GLY. В цепочках аминокислоты называются остатками - residues) и ещё пара параметров (A и 1), которые надо запомнить. Следующие три вещественных значения определяют координаты атома в молекуле.   
Далее еще пара чисел для запоминания (1 и 0) – это наличие и температурный фактор. Их не используем. В конце символ атома, по которому мы будем определять цвет и размер атома. Еще встречается параметр charge, который в этих файлах отсутствует, но переменную для него надо зарезервировать.

* Пробелы в файле имеют значение, поэтому считывать параметры надо по формату.  
  Класс атома с указанием позиций в строке  
    
  public class CAtom

{

public string stringname; // 01-06 "ATOM"

public int number; // 07-11 atom serial number

public string atomname; // 13-16 atom name

public char altLoc; // 17 alternate location indicator

public string residue; // 18-20 residue name

public string chain\_id; // 22 chain ID

public int nresidue; // 23-26 residue sequence number

public char iCode; // 27 code for insertion of residues

public float x; // 31-38 coord x

public float y; // 39-46 coord y

public float z; // 47-54 coord z

public float occupancy; // 55-60 occupancy

public float temp; // 61-66 temperature factor

public string symbol; // 77-78 symbol

public string charge; // 79-80 charge of the atom

}

Вещественные переменные в UNITY – float.

* **Разбор строки:**atom[count] = new СAtom();

atom[count].stringname = str.Substring(0, 6).Trim();

atom[count].number = Convert.ToInt32(str.Substring(6, 5).Trim());

atom[count].atomname = str.Substring(12, 4).Trim();

atom[count].altLoc = str.Substring(16, 1)[0];

atom[count].residue = str.Substring(17, 3).Trim();

atom[count].chain\_id = str.Substring(21, 1).Trim();

atom[count].nresidue = Convert.ToInt32(str.Substring(22, 4).Trim());

atom[count].iCode = str.Substring(26, 1)[0];

atom[count].x = (float)Convert.ToDouble(str.Substring(30, 8).Trim());

atom[count].y = (float)Convert.ToDouble(str.Substring(38, 8).Trim());

atom[count].z = (float)Convert.ToDouble(str.Substring(46, 8).Trim());

atom[count].occupancy = (float)Convert.ToDouble(str.Substring(54, 6).Trim());

atom[count].temp = (float)Convert.ToDouble(str.Substring(60, 6).Trim());

atom[count].symbol = str.Substring(76, 2);

atom[count].charge = " 0";

Разумеется, предварительно следует задать массив атомов.

* Для того, чтобы появились шарики атомов в местах, указанных в atom[count], надо ввести массив игровых объектов GameObject[] sphere:

sphere[iatom] = Instantiate(gsphere);

где gsphere – это объект класса GameObject

public GameObject gsphere;

gsphere = GameObject.Find("GeoSphere");

«GeoSphere» – это название объекта из Unity, который создаётся вручную:  
  


Другой способ получить шарик – это создать сферу программным путём:  
  
sphere[iatom] = GameObject.CreatePrimitive(PrimitiveType.Sphere);

Сферам следует придать нужные имя, цвет, размер, а также присвоить координаты:

**Имя:**

sphere[iatom].name = pdb[namino].atom[iatom].atomname;

**Цвет:**

Color color = Color.white;

float k = 0.25f;

string strColor = atom[iatom].symbol.Trim();

switch (strColor)

{

case "O": color = Color.Lerp(Color.red, Color.grey, k); break;

case "N": color = Color.Lerp(Color.blue, Color.grey, k); break;

case "C": color = Color.Lerp(Color.black, Color.grey, k \* 2f); break;

case "H": color = Color.Lerp(Color.white, Color.grey, k); break;

default:

color = Color.Lerp(Color.green, Color.green, k); break;

}

**Renderer** rend = sphere[iatom].GetComponent<Renderer>();

rend.material.color = color;

rend.material.shader = Shader.Find("Standard");

**Размер:**

sphere[iatom].transform.localScale = new Vector3(radius, radius, radius);

**Координаты:**

float x = - atom[iatom].x;

float y = atom[iatom].y;

float z = atom[iatom].z;

Vector3 newpos = new Vector3(x, y, z);

sphere[iatom].transform.position = newpos;

Работать с кучей атомов – вращать, двигать, не так удобно как с цельным объектом.   
Для этого в Unity есть опция родительского подчинения. На программном уровне это выглядит так:

public static GameObject handle = new GameObject();

sphere[iatom].transform.parent = handle.transform;

То есть, это пустой объект handle, к которому крепятся все сферы. Теперь этот объект представляет собой цельную молекулу – вращаем handle.

**Использование мышки:**

public Vector3 euler = new Vector3();

newpos.x = Input.mousePosition.x;

newpos.y = Input.mousePosition.y;

euler.x -=(newpos[0].x-oldpos[0].x)\*speed;

if (Input.GetKey(KeyCode.LeftShift))

{

euler.y -= 0f;

euler.z -= (newpos[0].y - oldpos[0].y) \* speed;

}

else

{

euler.y -= (newpos[0].y - oldpos[0].y) \* speed;

euler.z -= 0f;

}

Здесь добавлено вращение вокруг оси z при удержании LeftShift

Vector3 eulerv = new Vector3(-euler.z + euler\_old.z,   
 euler.x - euler\_old.x, euler.y - euler\_old.y);

**Вращение:**

handle[namino].transform.Rotate(eulerv, Space.World);

Вращать молекулу удобно с помощью движений мышки с нажатой левой кнопкой.  
Но, проще это сделать с помощью стрелок влево-вправо и вверх-вниз. Тут на выбор.

**Интерфейс**

Для выбора файла понадобится использовать возможности UNITY студии   
GameObject/UI/Canvas. Сделать меню и вывести в него названия всех молекул. По клику молекула отображается. Для этого надо использовать модуль OnGUI() и набор кнопок GUI.Button.

**Создание исполняемого файла**

Если проект компилируется без ошибок, то можно скомпилировать исполняемый файл.  
Для этого в UNITY студии выбираем  
File/Build Settings…, селектируем PC, Mac & Linux Standalone и жмём кнопку Build.  
  
**Удачи!**