snake version fenêtre 11/01: infos

Readme : /*

SNAKE codé par Sins en C++ avec la bibliothèque graphique SDL.
Date de début : Janvier

Le but du jeu:
Le but du jeu dans Snake est que le serpent mange toutes les pommes sans toucher les bords
de l'écran et sans se toucher lui-même.

Infos d'aide:
- Les pommes apparaîtront aléatoirement sur l'écran.
- le serpent bouge tout seul dans la direction dans laquelle il va,
On peut changer cette direction à l'aide des touches du clavier.
- le corps du serpent grossit d'une case à chaque fois qu'il mange une pomme.
- les bords de la fenêtre et le corps du serpent ne doivent pas pouvoir être traversés.

---

https://info.blaisepascal.fr/isn-projet-snake

https://github.com/shaswata56/Andropple

CODER PROPREMENT UN SNAKE
https://fr.sfml-dev.org/forums/index.php?topic=16285.0

on est a 8.15 secondes

sprite video
https://www.youtube.com/watch?v=aEDP7uhaiJc&ab_channel=Zenva

steave jobs (apple) dans le soleil des télétobbies + fond windows

perdu = blue screen
https://www.microsoft.com/en-ca/p/my-blue-screen/9wzdncrdr0sk?activetab=pivot:overviewtab#

http://atom.smasher.org/error/?icon=disk_skull&style=xp&title=Game+over&url=&text=T+o+o+++b+a+d&b1=Retry&b1g=x&b2=I%27m+a+looser&b3=

=> collision avec le bord
=> collision avec le corps
=> sprites de la pomme
=> sprites du snake
=> correction event
=> speed du snake

snake version fonction 11/01 :

deuxieme version
:::

```
#include <SFML/Graphics.hpp>
#include "objet.hpp"
#include <cstdlib>
//#include "fonction.hpp"


int main()

{

  int size = 40;
  int size_snake = 1;
  int speed = size;
  int var = 1;

  struct snake s[100];
  struct pomme p;
```

```
sf::RenderWindow window(sf::VideoMode(800,800),"SFML works!");
sf::RectangleShape shape(sf::Vector2f(size, size));
sf::RectangleShape pomme(sf::Vector2f(size,size));

/* sf::RectangleShape corps;
        corps.setSize(sf::Vector2f(size,size));
        corps.setOutlineColor(sf::Color::Black);*/



pomme.setFillColor(sf::Color::Green);

// shape.setOutlineThickness(10);
shape.setOutlineColor(sf::Color::Yellow);

//Limite de vitesse
window.setFramerateLimit(12);

//timer
// Clock timer;

//initialisation des positions de bases
p.x = 200;
p.y = 40;
s[0].x = 0;
s[0].y = 0;

pomme.setPosition(p.x,p.y);

shape.setPosition(s[0].x,s[0].y);


while(window.isOpen())

        {


        //  fin evennement

        sf::Event event;

        if(sf::Keyboard::isKeyPressed(sf::Keyboard::Escape))
  //(event.type == sf::Event::Close)
    {
        window.close();

    }
```

```
        // if(timer.getElapsedTime().asMilliseconds()>200)
//    {
  // Action();
  // timer.restart;
  //    }

        // Directions avec clavier
        if(sf::Keyboard::isKeyPressed(sf::Keyboard::Up)&& var != 2){var = 0;}
        if(sf::Keyboard::isKeyPressed(sf::Keyboard::Right)&& var != 3){var = 1;}
        if(sf::Keyboard::isKeyPressed(sf::Keyboard::Down)&& var != 0){var = 2;}
        if(sf::Keyboard::isKeyPressed(sf::Keyboard::Left)&& var != 1){ var = 3;}

        //deplacement(&var, &s, &size_snk);
        if (var == 0)
        {
        shape.move(0,-speed);
        s[0].y = s[0].y -speed;
        }
        if (var == 1)
        {
        shape.move(speed,0);
        s[0].x = s[0].x + speed;
        }
        if (var == 3)
        {
        shape.move(-speed,0);
        s[0].x = s[0].x - speed;
        }
        if (var == 2)
        {
        shape.move(0,speed);
        s[0].y = s[0].y + speed;
        }


                for(int i = size; i>0; i--)
        {
s[i].x = s[i-1].x;
s[i].y = s[i-1].y;
        }

        //Colision avec la pomme

        printf("s[0].x = %d, p.x = %d\n", s[0].x, p.x);
        printf("s[0].y = %d, p.y = %d\n\n", s[0].y, p.y);
        if(s[0].x == p.x && s[0].y == p.y)
```

```
        {
    size_snake++;
    p.x = rand()%(800/size);
    p.x = p.x * size;
    p.y = rand()%(800/size);
    p.y = p.y * size;

    pomme.setPosition(p.x, p.y);


        }
    // shape.setPosition(s[i].x,s[i].y);
    //   window.draw(pomme);
    // window.draw(shape) ;



    //    s[0].x && s[0].y == s[0].x && s[0].y + window.draw(corps) ;



        // Superposition avec shape et snake :

        window.clear();
        for ( int i = 0; i<size_snake;i++)
    {
      shape.setPosition(s[i].x,s[i].y);
      window.draw(pomme);
      window.draw(shape);

    }
        window.display();
        }

        return (0);  //test
}
```

11/01 MARIE :

```
#include <SFML/Graphics.hpp>

int main()

{

  int  x = 2;
  int  y = 5;
  sf::RenderWindow window(sf::VideoMode(900,900),"SFML works!");

  //SNAKE
  sf::RectangleShape shape(sf::Vector2f(40,50));

  shape.setFillColor(sf::Color::Green);
  window.setFramerateLimit(40);
  shape.setPosition(20,20);

  shape.setOutlineThickness(10);
  shape.setOutlineColor(sf::Color(250, 150, 100));

  //APPLE
  sf::CircleShape apple(50);
  apple.setFillColor(sf::Color(150, 50, 250));
  apple.setPosition(50,20);
  apple.setOutlineThickness(10);
  apple.setOutlineColor(sf::Color(250, 150, 100));
```

```
  while(window.isOpen())

        {

        sf::Event event;

        window.clear();

        window.draw(shape);
        window.draw(apple);
        window.display();

        }

  return 0;
}
```

https://en.sfml-dev.org/forums/index.php?action=search2

---

objet.hpp

```
#ifndef STRUCT_HPP_
#define STRUCT_HPP_

struct pomme

  {
  int  x = 2;
  int  y = 2;
  };

struct snake
  {
   int x,y;
  };

#endif
```

NEW VERSION AVEC POMME TEXTURE :

```
#include <SFML/Graphics.hpp>
#include "objet.hpp"
#include <cstdlib>
//#include "fonction.hpp"
#include <iostream>

int main()

{

  int size = 40;
  int size_snake = 1;
  int speed = size;
  int var = 1;

  struct snake s[100];
  struct pomme p;
```

```
sf::RenderWindow window(sf::VideoMode(800,800),"SFML works!");
sf::RectangleShape shape(sf::Vector2f(size, size));
sf::RectangleShape pomme(sf::Vector2f(size,size));

sf::Texture textureApple;
if (!textureApple.loadFromFile("assets/appleLogo.png")) {
  std::cout << "Erreur de chargement de la texture de la pomme.";
  return 0;
}
  sf::Sprite pommeSprite;
  pommeSprite.setTexture(textureApple);
  pommeSprite.setPosition(sf::Vector2f(1,1));
  pommeSprite.scale(sf::Vector2f(1,1));
/* sf::RectangleShape corps;
        corps.setSize(sf::Vector2f(size,size));
        corps.setOutlineColor(sf::Color::Black);*/



pomme.setFillColor(sf::Color::Green);

// shape.setOutlineThickness(10);
shape.setOutlineColor(sf::Color::Yellow);

//Limite de vitesse
window.setFramerateLimit(12);

//timer
// Clock timer;

//initialisation des positions de bases
p.x = 200;
p.y = 40;
s[0].x = 0;
s[0].y = 0;

pommeSprite.setPosition(p.x,p.y);

shape.setPosition(s[0].x,s[0].y);


while(window.isOpen())

        {


        //  fin evennement
```

```
        sf::Event event;

        if(sf::Keyboard::isKeyPressed(sf::Keyboard::Escape))
//(event.type == sf::Event::Close)
  {
        window.close();

  }


        // if(timer.getElapsedTime().asMilliseconds()>200)
//    {
  // Action();
  // timer.restart;
  //    }

        // Directions avec clavier
        if(sf::Keyboard::isKeyPressed(sf::Keyboard::Up)&& var != 2){var = 0;}
        if(sf::Keyboard::isKeyPressed(sf::Keyboard::Right)&& var != 3){var = 1;}
        if(sf::Keyboard::isKeyPressed(sf::Keyboard::Down)&& var != 0){var = 2;}
        if(sf::Keyboard::isKeyPressed(sf::Keyboard::Left)&& var != 1){ var = 3;}

        //deplacement(&var, &s, &size_snk);
        if (var == 0)
        {
        shape.move(0,-speed);
        s[0].y = s[0].y -speed;
        }
        if (var == 1)
        {
        shape.move(speed,0);
        s[0].x = s[0].x + speed;
        }
        if (var == 3)
        {
        shape.move(-speed,0);
        s[0].x = s[0].x - speed;
        }
        if (var == 2)
        {
        shape.move(0,speed);
        s[0].y = s[0].y + speed;
        }


                for(int i = size; i>0; i--)
        {
s[i].x = s[i-1].x;
```

```
    s[i].y = s[i-1].y;
          }

          //Colision avec la pomme

          printf("s[0].x = %d, p.x = %d\n", s[0].x, p.x);
          printf("s[0].y = %d, p.y = %d\n\n", s[0].y, p.y);
          if(s[0].x == p.x && s[0].y == p.y)
          {
    size_snake++;
    p.x = rand()%(800/size);
    p.x = p.x * size;
    p.y = rand()%(800/size);
    p.y = p.y * size;

    pommeSprite.setPosition(p.x, p.y);



          }
    // shape.setPosition(s[i].x,s[i].y);
    //   window.draw(pomme);
    // window.draw(shape) ;



    //    s[0].x && s[0].y == s[0].x && s[0].y + window.draw(corps) ;



          // Superposition avec shape et snake :

          window.clear();
          for ( int i = 0; i<size_snake;i++)
    {
      shape.setPosition(s[i].x,s[i].y);
      window.draw(pomme);
      window.draw(shape);
      window.draw(pommeSprite);

    }
          window.display();
          }

          return (0);
}
```

ORGANISATION :