Formation (BTS SIO SLAM – IRIS)

Snake Project

Membre du groupe:

Titouen
Aïssa
Liamidi Saobane

Next Slide

2025

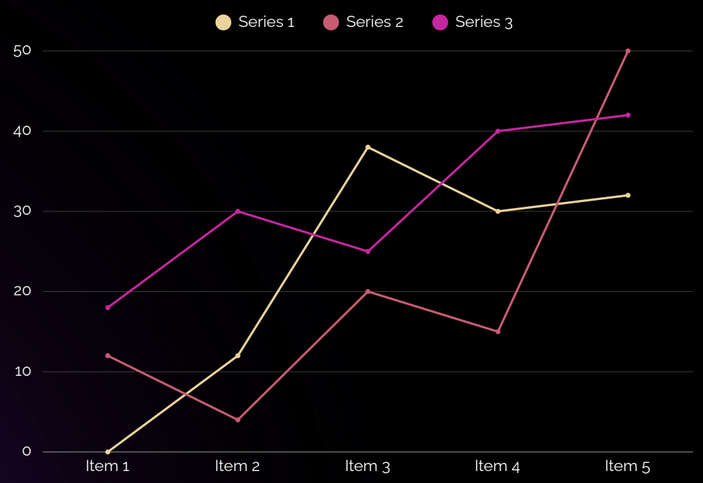
Sommaire

- Contexte & objectifs
- Présentation du groupe
- Cahier des charges du site
- Outils utilisés
- Organisation du travail
- Perspectives et conclusion

Next Slide

2025

Contexte & Objectif



Contexte et but du jeu

Le jeu Snake est un jeu vidéo classique apparu dans les années 1970, popularisé notamment sur les anciens téléphones Nokia.

Le principe est simple : le joueur contrôle un serpent qui se déplace dans un espace délimité. À chaque fois que le serpent mange une pomme (ou un autre élément), il grandit.

Le but du jeu est de manger un maximum de pommes sans se mordre la queue ni heurter les murs.

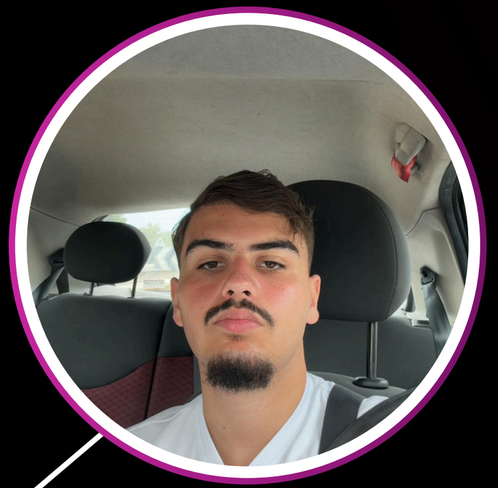
Dans le cadre de notre projet, le jeu a été revisité pour ajouter de nouvelles fonctionnalités :

- Génération de trois pommes à la fois ;
- Croissance du serpent uniquement après avoir mangé les trois ;
- Possibilité de sauvegarder et reprendre une partie ;
- Ajout de deux vitesses de jeu (normale et rapide).

Next Slide

Présentation du groupe

2025



Next Slide

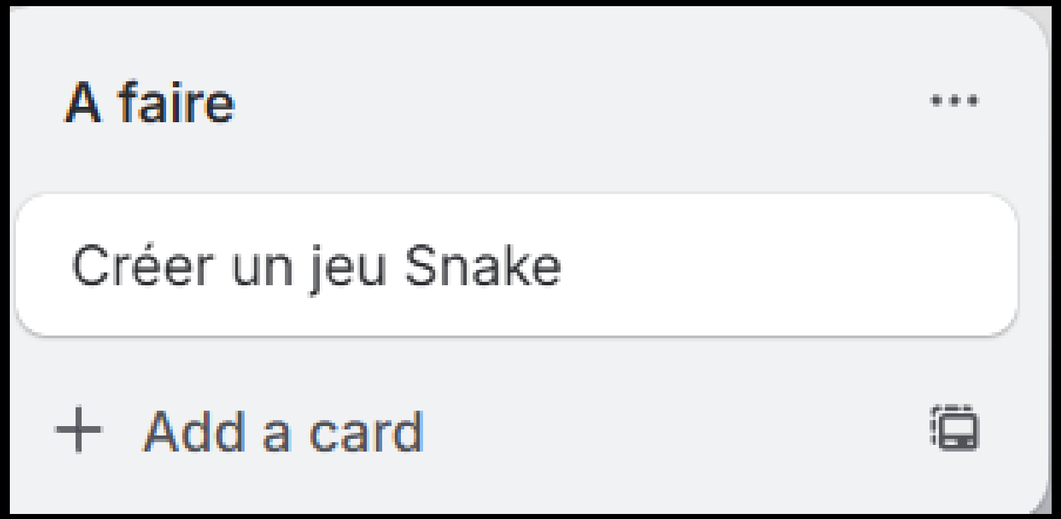
Cahier des charges

2025

Rubrique	**Contenu prévu**
Terminal	l’endroit du déroulement du jeu

Next Slide

Organisation du travail: Trello



Next Slide

Planning

2025

- Chercher le code source d’un snake déja fait
- Comprendre le code
- Trouver les parties à modifier
- Changer le code en fonction de la consigne demandé

Next Slide

Code source

2025

```
                    printf(" ");
                    len++;
                    if (len == length)
                        break;
                }
            }
        }
    }
    void Boarder()
    {
        system("cls");
        int i;

        // Afficher les 3 pommes
        for (i = 0; i < 3; i++)
        {
            if (food[i].x != 0 && food[i].y != 0) // Afficher seulement les pommes non mang�es
            {
                GotoXY(food[i].x, food[i].y);
                printf("P");
            }
        }

        for (i = 10; i < 71; i++)
        {
            GotoXY(i, 10);
            printf("!");
            GotoXY(i, 30);
            printf("!");
        }
        for (i = 10; i < 31; i++)
        {
            GotoXY(10, i);
            printf("!");
            GotoXY(70, i);
```

```
    int Score()
    {
        int score;
        GotoXY(20, 8);
        score = length - 5;
        printf("Score : %d", (length - 5));
        score = length - 5;
        GotoXY(50, 8);
        printf("Vies : %d", life);
        return score;
    }
    int Scoreonly()
    {
        int score = Score();
        system("cls");
        return score;
    }
    void Up()
    {
        int i;
        for (i = 0; i <= (bend[bend_no].y - head.y) && len < length; i++
        {
            GotoXY(head.x, head.y + i);
            {
                if (len == 0)
                    printf("^");
                else
                    printf("*");
            }
            body[len].x = head.x;
            body[len].y = head.y + i;
            len++;
        }
        Bend();
        if (!kbhit())
            head.y--;
    }
```

```
                for (j = i; j <= diff; j++)
                    /*GotoXY(bend[i].x,(bend[i].y-j));
                    printf("*");*/
                    body[len].x = bend[i].x;
                    body[len].y = bend[i].y - j;
                    GotoXY(body[len].x, body[len].y);
                    printf("*");
                    len++;
                    if (len == length)
                        break;
                }
            }
            else if (bend[i].y == bend[i - 1].y)
            {
                diff = bend[i].x - bend[i - 1].x;
                if (diff < 0)
                    for (j = i; j <= (-diff) && len < length; j++)
                    {
                        /*GotoXY((bend[i].x+j),bend[i].y);
                        printf("*");*/
                        body[len].x = bend[i].x + j;
                        body[len].y = bend[i].y;
                        GotoXY(body[len].x, body[len].y);
                        printf("*");
                        len++;
                        if (len == length)
                            break;
                    }
                else if (diff > 0)
                    for (j = i; j <= diff && len < length; j++)
                    {
                        /*GotoXY((bend[i].x-j),bend[i].y);
                        printf("*");*/
                        body[len].x = bend[i].x - j;
                        body[len].y = bend[i].y;
                        GotoXY(body[len].x, body[len].y);
```

```
        for (i = 4; i < length; i++) //starts with 4 because it needs minimum 4 element to touch its own body
        if (head.x <= 10 || head.x >= 70 || head.y <= 10 || head.y >= 30 || check != 0)
        {
            life--;
            if (life >= 0)
            {
                head.x = 25;
                head.y = 20;
                bend_no = 0;
                head.direction = RIGHT;
                Move();
            }
            else
            {
                system("cls");
                printf("Vous etes mort !\nAppuyez sur une touche pour quitter\n");
                record();
                exit(0);
            }
        }
    }
    void Food()
    {
        int i;
        int allEaten = 1;

        // V�rifier si toutes les pommes ont �t� mang�es
        for (i = 0; i < 3; i++)
        {
            if (food[i].x != 0 && food[i].y != 0) // Si une pomme existe encore
            {
                allEaten = 0;
                break;
            }
        }

        // Si toutes les pommes sont mang�es, reg�n�rer les 3 pommes et agrandir le serpent
        if (allEaten == 1)
```

```
        Left();
    else if (head.direction == DOWN)
        Down();
    else if (head.direction == UP)
        Up();
    ExitGame();
} while (!kbhit());
a = getch();
if (a == 27)
{
    system("cls");
    exit(0);
}
key = getch();
if ((key == RIGHT && head.direction != LEFT && head.direction != RIGHT) || (key == LEFT && head.direct
{
    bend_no++;
    bend[bend_no] = head;
    head.direction = key;
```

```
    {
        length++;
        foodsEaten = 0;
        for (i = 0; i < 3; i++)
        {
            food[i].x = rand() % 70;
            if (food[i].x <= 10)
                food[i].x += 11;
            food[i].y = rand() % 30;
            if (food[i].y <= 10)
                food[i].y += 11;
        }
    }

    // V�rifier les collisions avec chaque pomme
    for (i = 0; i < 3; i++)
    {
        if (head.x == food[i].x && head.y == food[i].y)
        {
            food[i].x = 0; // Marquer la pomme comme mang�e
            food[i].y = 0;
            foodsEaten++;
            break;
        }
    }

    // G�n�ration initiale des pommes (seulement au d�but du jeu)
    static int firstTime = 1;
    if (firstTime)
    {
        firstTime = 0;
        for (i = 0; i < 3; i++)
        {
            food[i].x = rand() % 70;
            if (food[i].x <= 10)
                food[i].x += 11;
            food[i].y = rand() % 30;
```

```
    void Food()
    {
        if (firstTime)
            for (i = 0; i < 3; i++)
                food[i].y = rand() % 30;
                if (food[i].y <= 10)
                    food[i].y += 11;
            }
        }
    }
    void Left()
    {
        int i;
        for (i = 0; i <= (bend[bend_no].x - head.x) && len < length; i++)
        {
            GotoXY((head.x + i), head.y);
            {
                if (len == 0)
                    printf("<");
                else
                    printf("*");
            }
            body[len].x = head.x + i;
            body[len].y = head.y;
            len++;
        }
        Bend();
        if (!kbhit())
            head.x--;
    }
    void Right()
    {
        int i;
        for (i = 0; i <= (head.x - bend[bend_no].x) && len < length; i++)
        {
            //GotoXY((head.x-i),head.y);
            body[len].x = head.x - i;
            body[len].y = head.y;
            GotoXY(body[len].x, body[len].y);
```

```
    void Down()
    {
        int i;
        for (i = 0; i <= (head.y - bend[bend_no].y) && len < length; i++)
        {
            GotoXY(head.x, head.y - i);
            {
                if (len == 0)
                    printf("v");
                else
                    printf("*");
            }
            body[len].x = head.x;
            body[len].y = head.y - i;
            len++;
        }
        Bend();
        if (!kbhit())
            head.y++;
    }
    void Delay(long double k)
    {
        Score();
        long double i;
        for (i = 0; i <= (10000000); i++)
            ;
    }
    void ExitGame()
    {
        int i, check = 0;
        for (i = 4; i < length; i++) //starts with 4 because it needs minimum 4 element to touch its own body
        {
            if (body[0].x == body[i].x && body[0].y == body[i].y)
            {
                check++; //check's value increases as the coordinates of head is equal to any other body coor
            }
            if (i == length || check != 0)
                break;
        }
```

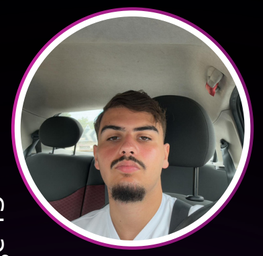

Next Slide

2025

Bilan personnel

Ce projet m’a permis de renforcer ma compréhension de la logique algorithmique et de la programmation en C. J’ai appris à structurer un programme complet, à gérer les déplacements du serpent et la détection des collisions. J’ai également découvert l’importance de la rigueur dans la gestion de la mémoire et du débogage. Ce travail m’a aidé à progresser dans la conception de programmes interactifs et à mieux comprendre le fonctionnement des boucles et conditions imbriquées.

Grâce à ce projet, j’ai développé mes compétences en travail d’équipe et en organisation du code. J’ai participé à la mise en place des différentes fonctionnalités du jeu, notamment la génération des pommes et la gestion du score. J’ai appris à lire et modifier un code existant pour l’adapter aux nouvelles consignes, ce qui m’a permis de gagner en autonomie et en efficacité dans mes approches de résolution de problèmes.



Ce projet m’a permis de mieux comprendre le fonctionnement d’un jeu vidéo simple en langage C. J’ai travaillé sur la partie affichage et sur l’adaptation du programme aux nouvelles fonctionnalités, comme la sauvegarde et la reprise d’une partie. J’ai également découvert l’intérêt de bibliothèques externes comme conio.h ou SDL pour gérer l’affichage et les interactions. Cette expérience m’a montré l’importance du travail collaboratif et du partage de connaissances dans la réussite d’un projet.

Next Slide

Conclusion

Ce projet Snake nous a permis d’appliquer nos connaissances en langage C et en algorithmique. Nous avons appris à analyser et modifier un programme existant, à ajouter de nouvelles fonctionnalités et à collaborer efficacement en équipe. C’était une expérience enrichissante sur les plans technique et pratique.

Next Slide

2025

ThankYou!