INTERNATIONAL BINA NUSANTARA

SNEK GAME

Wilson Fransicius Willyam Lo

2001585303

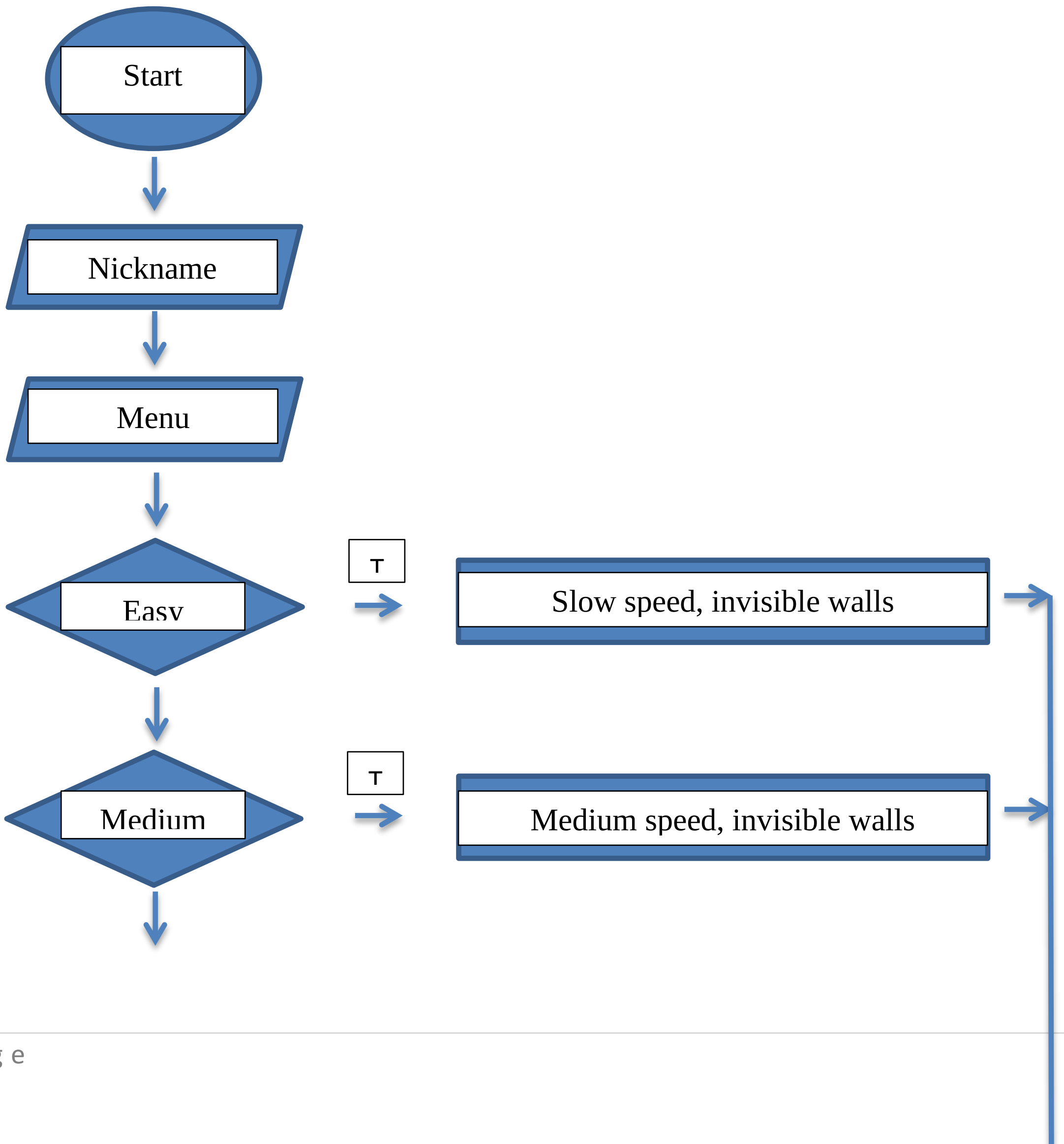
06/Nov/2016

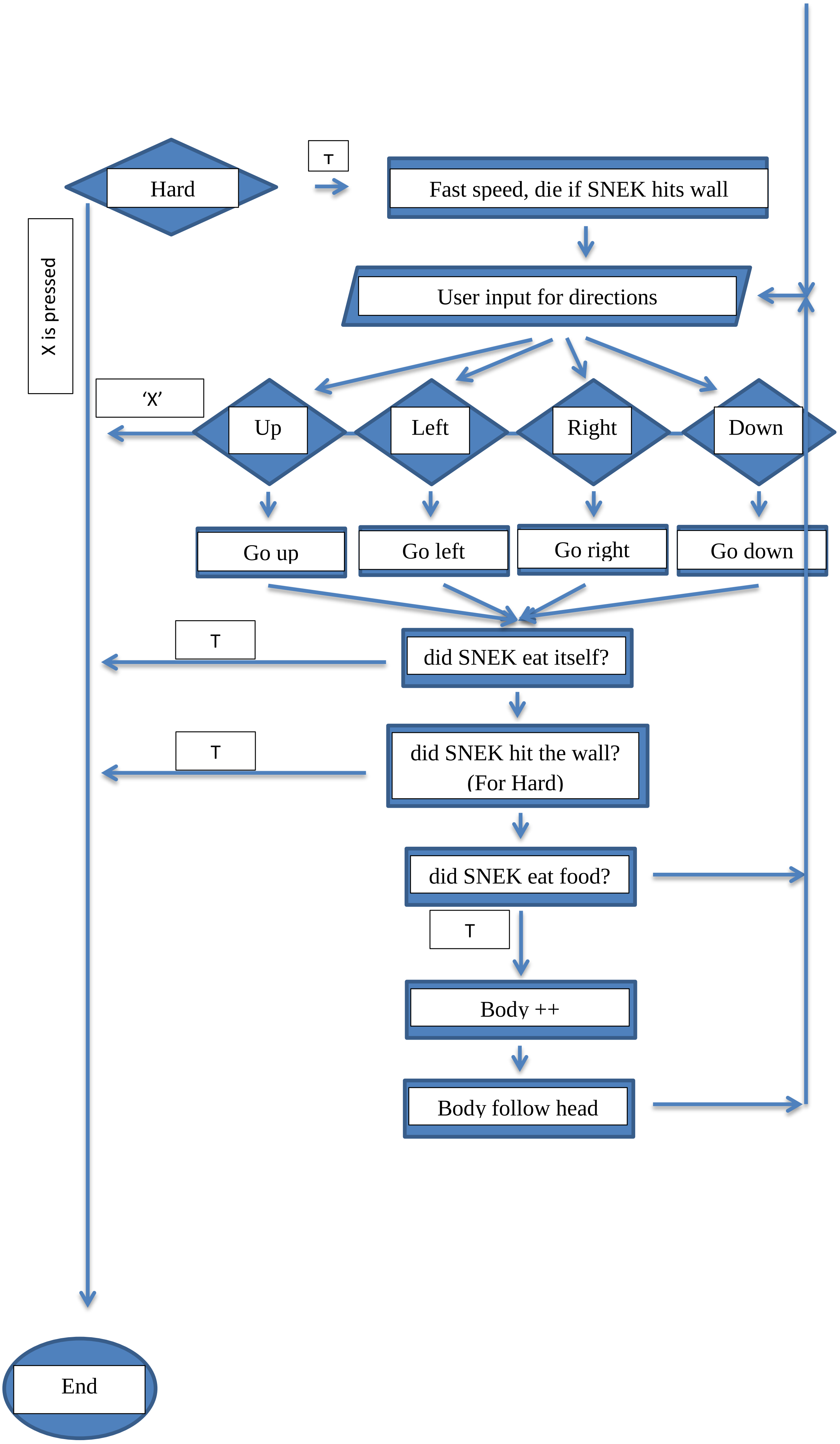
A classic snake game written in C++

I. Description

Everyone know what is Snake, especially if you had Nokia phone around 2000s, SNEK is similar to the classic Nokia Snake which was a single player game that require the player to beat the highest score which has been achieved either by other player or the non-player character (NPC). Instead of beating the highest score, SNEK does not even save the score player has achieved, this has a purpose, which is to make sure that players did not forget to study instead of playing SNEK all day long. The game itself is simple, first user has to input their nickname and enter the mode they want to play (easy, medium, or hard) then the game run, after the game is over, it will show much score does player has achieved and a comment.

II. Design & Plan




Hard
T
Fast speed, die if SNEK hits wall
User input for directions
X is pressed
'X'
Up
Left
Right
Down
Go up
Go left
Go right
Go down
T
did SNEK eat itself?
T
did SNEK hit the wall? (For Hard)
did SNEK eat food?
T
Body ++
Body follow head
End

III. Explanation of Functions

Int main()

It is the main function that reads and run all the function below it also asks the user to input their nickname, output the score, and output the reminder to study and not playing, and ask whether the user wants to play again.

Void Menu()

It contains interface, the interface itself will ask the difficulty the user wants whether it’s easy, medium or hard. After user inputs the difficulty, menu will call Base(), Cage(), Input(), and Body(), and will be run in main function.

Void Base()

This is the basic setup for the entire game, it contains the start location(x,y) and the first food location (x,y).

Void Cage()

This is where the cage is drawn, and the start head of SNEK and the food, the cage is drawn with “#” using loop.

Void Input()

This is the function which takes the input from the user, it uses *_kbhit*, and *_getch* for the keyboard input, and use switch case for the logic and direction.

Void Body()

This is the most important function because it contains most of the logic for SNEK body, to make sure the body follow its head, array is used, and for the direction in Input(), the logic is written here, also if SNEK eats its food, score increment by five, and if SNEK eats its own tail, game over is true.

IV. What I have learnt and problems I have overcome

Through this project I have learnt to apply things that have been told by my facilitators, especially if statements logic and array, I also learnt many new things such as *system clear*, *_kbhit*, *_getch*, and *sleep*. I had faced many problems, but I have overcome it, special thanks to Jeffrey, Dylan, and Archel for helping me by answering my questions and explain where I was wrong, to make it easier, this is my log and things I had done throughout the month :

07/10/2016	Planning phase for classic tamagotchi game
11/10/2016	New idea, planning phase for classic snake game
14/10/2016	Cage
21/10/2016	Cage Improvement + eNum Direction + Head + Food
28/10/2016	Moving everything to class + input kbhit + switch + length++ when hit food
04/11/2016	Cancle the class because program won't run bool failed
05/11/2016	Create array to make the body follow its head + add difficuty + seperate the functions into functions.hpp

On the second week, I was afraid that I could not make classic Tamagotchi game because I thought that it is hard make the moving pixels interface, so instead of Tamagotchi, I create SNEK. On Friday, I started to make it, so I created the hierarchy and flow charts, my first progress was to make the cage that the snake should not hit.

On the third week, I Improved the cage because the previous cage was not symmetrical, then I added the direction with eNum.

On the fourth week, I tried using class, I also created the input with *kbhit* and *getch* which I had googled and I learnt new library which is <conio.h>, and I also stated a logic that if head hit food, food should generate a new food.

On the fifth week, when I run my program, it worked completely fine, but when it hit the wall, the game does not over, instead the snake disappeared, to be honest instead of overcoming the problem, I delete the class, since it was almost due date I did not want to take the risk. And for the last part, which was the hardest part was to make the body of the snake follow its head, but thankfully I overcame it by using array, so I stored the previous location in array and loop it.

V. Coding

main.cpp

```
#include "Function.hpp"
char name[100];

int main()
{

        cout << "Enter Nickname: ";
        cin.getline (name, 100);
        Menu();

        system ("cls");
        cout << "Congrats "<< name << ", your Score is: " << score<< endl << endl;

        if(score>=0 && score<=20)
        {
                cout << "\nThis game is hard, why don't give it one more try? :)";
        }
        else if (score>20 && score <=35)
        {
                cout << "\nPlaying SNEK is fun, but don't forget to study!";
        }
        else if (score>35 && score <=55)
        {
                cout << "\nWhoaa, we have a pro here, enough playing, go study!";
        }
        else if (score>=55 && score <=100)
        {
                cout << "\nGODLIKE";
        }
        else
        {
                cout << "ENOUGH! NOW STUDY!!!";
        }

        return 0;
```

Function.cpp

```
#ifndef Function_hpp
#define Function_hpp
#include <iostream>
#include <cstdlib>              // for rand()
#include <conio.h>              // user input (kbhit, getch)
#ifdef _WIN32
#include <Windows.h>            // for windows user (sleep)
#else
#include <unistd.h>             // for other than windows user
#endif
using namespace std;
int x, y, foodX, foodY, score;
int width = 75;
int height = 20;
bool gameOver;
enum direction {START=0, LEFT, RIGHT, UP, DOWN};
direction dir;
int arrX[100], arrY[100], body;
```

```
/*
void Base();   //basic setup
void Cage();   //cage
void Body();   //snake body
void Input();  //user input (w,a,s,d)
*/
void Base()
{
        gameOver = false;
        dir = START;                          //START = 0 = middle of the cage
        x = width/2;                          //start position x for snake
        y = height/2;                         //start position y for snake
        foodX = rand() % width;               //Generate random food X coordinate
        foodY = rand() % height;              //Generate random food Y coordinate
}

void Cage()
{
        system("cls"); // clear screen
        cout << " ";
        for (int i = 0; i < width+1; i++)            // first row
        {
                cout << "#";
        }
        cout << endl << " ";   // to seperate & and X
        for (int k = 0; k < height; k++)      //Y
        {
                for (int l=0; l < width; l++) //X
                {
                        if (l ==0)     //left column
                        {
                                cout << "#";
                        }
                        if (l == width-1) // right column
                        {
                                cout << "#" << endl;
                        }
                        if (l == x && k == y) // snake head logic
                        {
                                cout << "S";
                        }
                        else if (l == foodX && k == foodY)    //random food logic
                        {
                                cout << "O";
                        }

                }
        }
        for (int j = 0; j < width+1; j++)     // last row
        {
                cout << "#";
        }
        cout << endl << " Score: "<< score << endl;
        cout << "---------- Press [X] to Exit ---------" << endl;
}

void Body()
{
        int prevX = arrX[0];   // to remember the before body parts of snake before eat
        int prevY = arrY[0];
        int prev2X, prev2Y;
        arrX[0] = x;           // to make the body follow its head
        arrY[0] = y;
        for (int i=1; i < body; i++)
        {
                prev2X = arrX[i];             // the 2 prev body follow the prev body
                prev2Y = arrY[i];
                arrX[i] = prevX;              // change(update) its value to prevX
                arrY[i] = prevY;
                prevX = prev2X;                       // prev X to follow the head
                prevY = prev2Y;
        }
        switch(dir)
        {
                case LEFT:
                        x--;
                        break;
                case RIGHT:
                        x++;
                        break;
                case UP:
                        y--;    // decrease the y value
                        break;
                case DOWN:
                        y++;    // increase the y value
                        break;
                default:
                        break;
        }
        for (int i = 0; i < body; i++)// to make it's game over when snake eat itself
        {
                if (arrX[i] == x && arrY[i]==y)
                {
                        gameOver = true;
                }
        }
        if (x == foodX && y == foodY) // body ++ when eat food
        {
                body++;
                score += 5;
                foodX = rand() % width;         // new food x
                foodY = rand() % height;        // new food y
        }
        else
        {
                bool follow = false;
                for (int m=0; m<body; m++)      //to make the body follow its head
                {
                        if (arrX[m] == l && arrY[m] == k)      // loop to make body ++ when
                        {
                                cout << "o";
                                follow = true;
                        }
                }
                if(!follow)     //if the body follows the head, it should print blank space
                {
                        cout << " ";
                }
        }
}

void Input()
{
        if (_kbhit())  //determines if a keyboard was pressed
        {
                switch (_getch())       //reads a charcter from keyboard
                {
                        case 'a' :
                                dir = LEFT;
                                break;
                        case 'd':
                                dir = RIGHT;
                                break;
                        case 'w':
                                dir = UP;
                                break;
                        case 's':
                                dir = DOWN;
                                break;
                        case 'x':               // press x to terminate the snek
                                gameOver=true;
                                break;
                        default:
                                break;
                }
        }

}
```

```
void Menu()
{
        char input1;
        string input2;

        do
        {
                system("cls");
                cout << "                     #          #  " << endl;
                cout << " ##### #####   ##### #    #  " << endl;
                cout << "#      #    # #   #  #   #   " << endl;
                cout << " ####  #    # #  #   # ##    " << endl;
                cout << "     # #    # #      #   #   " << endl;
                cout << "#####  #    #  ##### #    #  " << endl << endl << endl;
                cout << "Press [X] to Continue" << endl;
                cin >> input1;

        }
        while ((input1!='x') && (input1!='X'));

        do
        {
                system("cls");
                cout << "---------------------------------------" << endl;
                cout << "---------- Welcome to SNEK! ----------" << endl;
                cout << "--- Press [1] to Play In Easy Mode ---" << endl;
                cout << "-- Press [2] to Play In Medium Mode --" << endl;
                cout << "--- Press [3] to Play In Hard Mode ---" << endl;
                cout << "---------- Press [X] to Exit ---------" << endl;
                cout << "---------------------------------------" << endl;
                cin >> input2;
                if (input2 == "1")
                {
                        system("cls");
                        Base();
                        do
                        {
                                Cage();
                                Input();
                                if (x >= width) x = 0; else if (x < 0) x = width - 1;
                                // snek go through the walls
                        if (y >= height) y = 0; else if (y < 0) y = height - 1;
                                Body();

                                Sleep(30);
                        }
                        while(!gameOver);
                }


                else if (input2 == "2")
                {
                        system("cls");
                        Base();
                        do
                        {
                                Cage();
                                Input();
                                if (x >= width) x = 0; else if (x < 0) x = width - 1;
                        if (y >= height) y = 0; else if (y < 0) y = height - 1;
                                Body();

                                Sleep(10);
                        }
                        while(!gameOver);
                }
                else if (input2 == "3")
                {
                        system("cls");
                        Base();
                        do
                        {
                                Cage();
                                Input();
                                Body();
                                if (x > width+1 || x < 0 || y > height || y < 0)
// to make it's game Over when snake hits the cage
                        {
                                        gameOver = true;
                                }
                                Sleep(-50);
                        }
                        while(!gameOver);
                }
                else if (input2 == "x" || input2 =="X")
                {
                        gameOver=true;
                }
        }
while ((input2!="1") && (input2!="2") && (input2!="3") && (input2!="X") && (input2!="x"));
}
#endif
```