
# “Snake Game”

## Using Stack

**Ansari Arif(3117001)**

**Memon Shaziya(3117026)**

**Mishra Saloni(3117027)**

**Second Year Computer Engineering(Sem-III)**

Supervisor

**Prof. Waheeda Dhokley**

**Department of Computer Engineering**

**M. H. Saboo Siddik College of Engineering**

**8,Saboo Siddik Polytechnic Road, New Nagpada, Byculla,**

**Mumbai,Maharashtra 400008**

**2018**

# Problem Definition:

## Program to implement the snake game using stack data structure.

## Introduction :

This **Mini Project in C Snake Game** is a simple console application without graphics. In this project, you can play the popular “Snake Game” just like you played it elsewhere. You have to use the up, down, right or left arrows to move the snake.

## Explaination :

This project aims to bring the fun and simplicity of snake. Mini project in C **Snake Game** .Foods are provided at the several co-ordinates of the screen for the snake to eat. Every time the snake eats the food, its length will by increased by one element along with the score.

The source code for **Snake Game in C** is complete and totally error-free. It is compiled in Code::blocks using the gcc compiler. The code is about 550 lines; so I haven’t displayed the source code here. You can directly download the source code plus application file from the link below.

# Functions used in Snake Game Project in C:

Many functions have been used in this Snake mini project. Here, I will just list them below and describe the functions “gotoxy”, “GotoXY” and “delay” as they are some of the most important functions used in this and many mini projects in C.

void record()
void load()
void Delay(long double)
void Move()
void Food()
void Print()
void Bend()
int Score()
void Boarder()
void Down()
void Left()
void Up()
void Right()
void ExitGame()

**void gotoxy (int x, int y)** – You need to understand this function as it is one of the most important one used in Snake Game mini project in C. This function allows you to print text in any place of screen. Using this function in Code::Blocks requires coding, but it can be directly used in Turbo C. Here is a code for this function in Code::Blocks.

Code for gotoxy (Mini Project in C Snake Game)

Here, *COORD coord= {0,0};* is a global variable. It sets the center of axis to the top left corner of the screen.

**void GotoXY (int x, int y)** – Here is the code for this function in Code::Blocks.

Code for GotoXY (Mini Project in C Snake Game)

```
1  void GotoXY(int x, int y)
2  {
3      HANDLE a;
4      COORD b;
5      fflush(stdout);
6      b.X = x;
7      b.Y = y;
8      a = GetStdHandle(STD_OUTPUT_HANDLE);
9      SetConsoleCursorPosition(a,b);
10 }
```

**void delay(long double)** – This function delays the execution. It can be used directly in Turbo C, but requires coding in Code::Blocks. The code is given below:

Code for delay (Mini Project in C Snake Game)

```
1 void Delay(long double k)
2 {
3    Score();
4    long double i;
5    for(i=0; i<=(10000000); i++);
6 }
```

This mini project in C Snake game gives users a total of three lives to play the game. The life-count decreases as the snake hits the wall or its own body. In this mini project, you can even pause the game in the middle by pressing any key, and you can press any key again to continue.

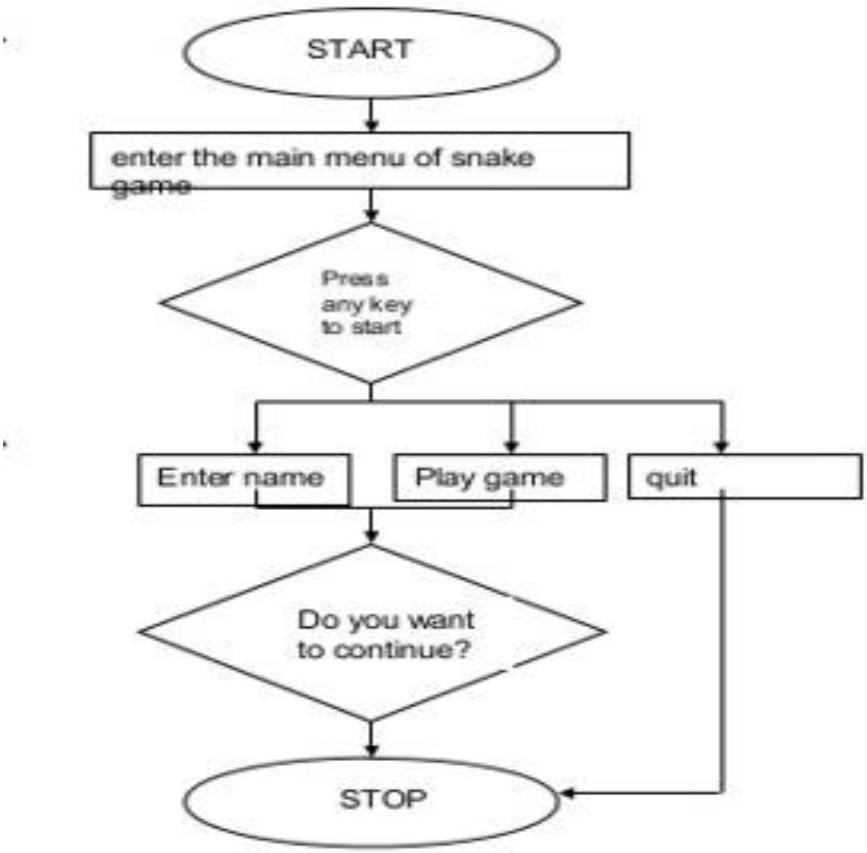
## Game Instructions:

To sum it up, this mini project on Snake game in C allows you to record the player's name and the corresponding score obtained. File handling has been used for that purpose.

This mini project can definitely help you if you were looking for a reference project or looking to create a C mini project game of your own. Submitting this project with little or no modification at all is completely discouraged.

# SYSTEM DESIGN:

Then we began with the design phase of the system. System design is a solution, a "HOWTO" approach to the creation of a new system. It translates system requirements intoways by which they can be made operational. It is a translational from a user orienteddocument to a document oriented programmers. For that, it provides the understandingand procedural details necessary for the impleme ntation. Here we use Flowchart tosupplement the working of the new system. The system thus made should be reliable,durable and above all should have least possible maintenance costs. It should overcomeall the drawbacks of the Old existing system and most important of all meet the user requirements.

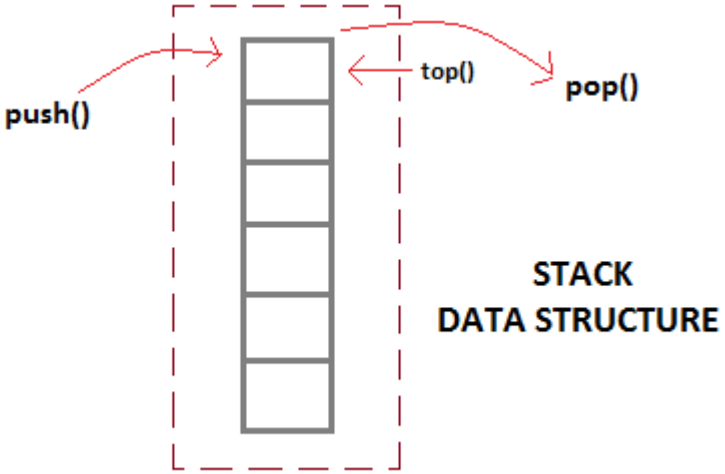



In these program,  we are using stack in data structure.

Stack:

# What is Stack Data Structure?

**Stack** is an abstract data type with a bounded(predefined) capacity. It is a simple data structure that allows adding and removing elements in a particular order. Every time an element is added, it goes on the **top** of the stack and the only element that can be removed is the element that is at the top of the stack, just like a pile of objects.




---

# Basic features of Stack

1. Stack is an **ordered list** of **similar data type**.
2. Stack is a **LIFO**(Last in First out) structure or we can say **FILO**(First in Last out).
3. `push()` function is used to insert new elements into the Stack and `pop()` function is used to remove an element from the stack. Both insertion and removal are allowed at only one end of Stack called **Top**.
4. Stack is said to be in **Overflow** state when it is completely full and is said to be in **Underflow** state if it is completely empty.

---

# Applications of Stack

The simplest application of a stack is to reverse a word. You push a given word to stack - letter by letter - and then pop letters from the stack.

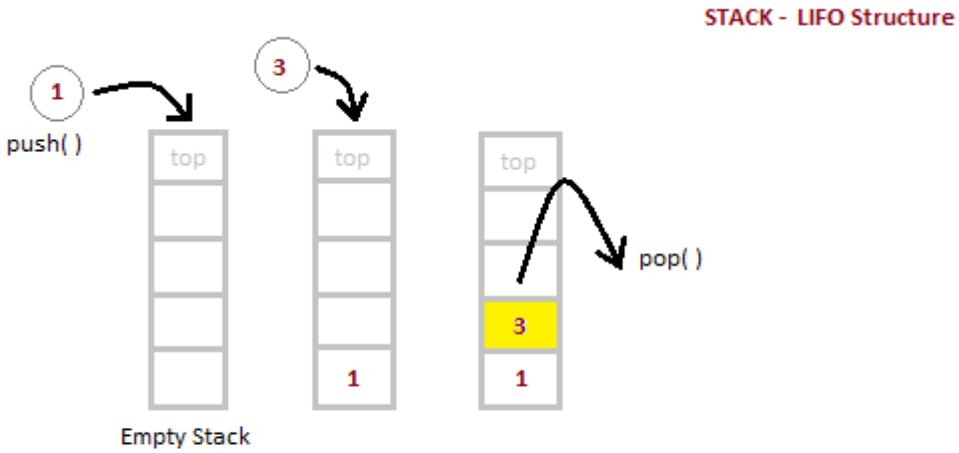
There are other uses also like:

1. Parsing
2. Expression Conversion(Infix to Postfix, Postfix to Prefix etc)

---

# Implementation of Stack Data Structure

Stack can be easily implemented using an Array or a Linked List. Arrays are quick, but are limited in size and Linked List requires overhead to allocate, link, unlink, and deallocate, but is not limited in size. Here we will implement Stack using array.




In a Stack, all operations take place at the "**top**" of the stack. The "**push**" operation adds an item to the top of the Stack.
The "**pop**" operation removes the item on top of the stack.