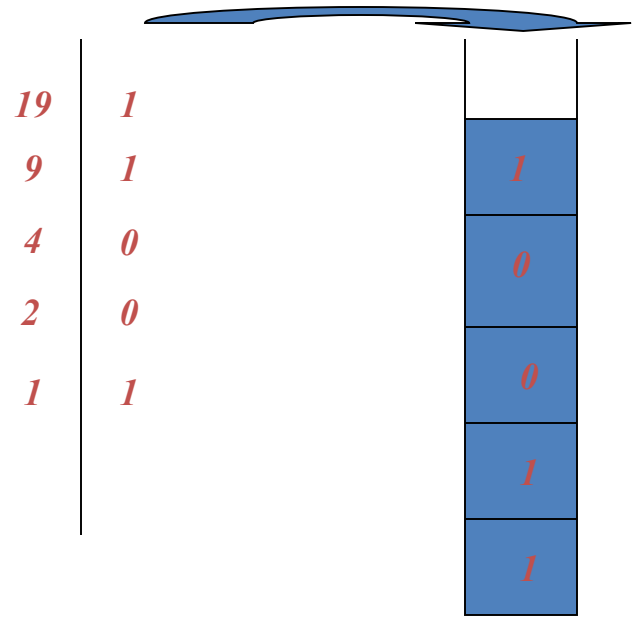
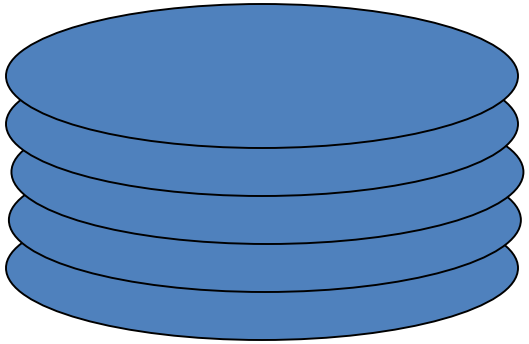
Ngăn xếp (Stack)

Ngăn xếp (Stack)

- Là cấu trúc bao gồm các phần tử được truy xuất theo nguyên tắc “vào sau, ra trước” (Last In, First Out – LIFO)

  Ví dụ: Chồng đĩa, đổi số ra nhị phân





Các thao tác trên Stack

4 thao tác cơ bản trên stack

- Khởi tạo stack rỗng: ***StackA(int s)***
- Kiểm tra stack rỗng: ***boolean isEmpty()***
- Đưa phần tử vào Stack: ***push(int j)***
- Lấy phần tử ra khỏi Stack: ***int pop()***

2 thao tác hỗ trợ trên stack

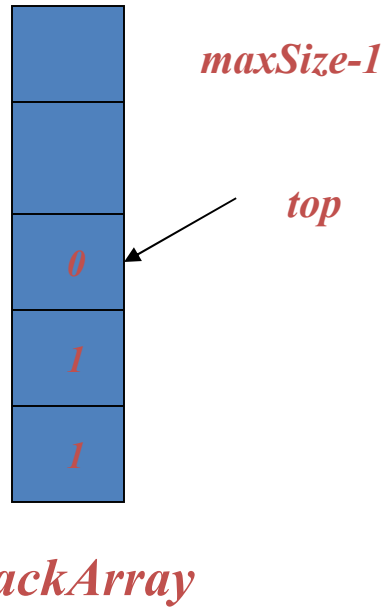
- Lấy giá trị trên đỉnh stack: ***int peek()***
- Kiểm tra stack s đầy?: ***boolean isFull()***

Cài đặt trên cơ sở mảng

Sử dụng:

- Mảng ***stackArray[maxSize]*** phần tử
- Chỉ số ***top*** để chỉ đỉnh stack (nơi đưa vào, lấy ra)

```
class StackA {
    private int maxSize;  // size of stack array
    private int[] stackArray;
    private int top;              // top of stack
            ……
}
```



Các thao tác trên Stack

- **Khởi tạo stack rỗng có kích thước s**

```
public StackA(int s)                          // constructor
{
      maxSize = s;                            // set array size
      stackArray = new int[maxSize];          // create array
      top = -1;                               // no items yet
}
```

Kiểm tra stack s rỗng?

```
public boolean isEmpty() // true if stack is empty
{
      return (top == -1);
}
```

Các thao tác trên Stack

- **Đưa phần tử j vào stack s**

```
public void push(int j)                 // put item on top of stack
 {
      stackArray[++top] = j;            // increment top, insert item
 }
```

Lấy phần tử j ra khỏi stack s

```
public int pop()                        // take item from top of stack
 {
      return stackArray[top--];         // access item, decrement top
 }
```

Các thao tác trên Stack

Lấy giá trị trên đỉnh stack

```
public int peek() // peek at top of stack
{
        return stackArray[top];
}
```

Kiểm tra stack s đầy?

```
public boolean isFull()                    // true if stack is
  full
{
        return (top == maxSize-1);
}
```

Ví dụ: Đổi số ra nhị phân

```
class Stack
{
   public static void main(String[] args)
   {
      StackA s = new StackA(50);
      int n, i;
      Scanner x;
      x = new Scanner(System.in);
      System.out.print("Nhap n:");
      n = x.nextInt();
      while (n>0)
      {
         i = n%2;
         s.push(i);
         n = n/2;
      }
      System.out.print("Dang nhi phan:");
      while (!s.isEmpty())
      {
         i = s.pop();
         System.out.print(""+i);

      }
      System.out.println();

   }
}
```

Đệ qui và tổ chức đệ qui

- Một định nghĩa được gọi là đệ qui nếu nó được định nghĩa trên chính nó một cách trực tiếp hay gián tiếp
- Đệ qui luôn gồm 2 phần
  - Phần dừng
  - Phần đệ qui

Ví dụ

$$x^n = \begin{cases} 1 & n = 0 \\ x.x^{n-1} & n > 0 \end{cases}$$

$$n! = \begin{cases} 1 & n = 0 \\ n(n-1)! & n > 0 \end{cases}$$

$$US(a,b) = \begin{cases} a & b = 0 \\ US(b, a\%b) & b > 0 \end{cases}$$

Giai thừa đệ qui

```
public class Giathua {
        static long gt(int n)
        {
                if (n==0) return 1l;
                return n*gt(n-1);
        }
        public static void main(String[] args)
        {
                System.out.print(""+gt(5));
        }
}
```

Khử bỏ đệ qui

- Sử dụng Stack
- Đệ qui được thay bằng:
  - Hàm đệ qui: Vòng lặp
  - Lời gọi đệ qui: Push các giá trị cục bộ
  - Thực hiện đệ qui: Pop các giá trị cục bộ
- Lưu ý: Stack là cấu trúc LIFO

Bài toán Tháp Hanoi đệ qui

```
public class BTThapHN
{
	static void chuyen(int n, char a, char b, char c)
	{
		if (n==1)
			System.out.println("Chuyen tu " + a + " qua " + c);
		else
		{
			chuyen(n-1, a, c, b);
			chuyen(1, a, ' ', c);
			chuyen(n-1, b, a, c);
		}
	}
	public static void main(String[] args)
	{
		chuyen(3, 'a', 'b', 'c');
	}
}
```

Bài toán Tháp Hanoi không đệ qui

1. Khởi tạo stack s rỗng
2. Push 1 bộ (3, ‘A’, ‘B’, ‘C’) vào s
3. Lặp
   - Pop ra khỏi s 1 bộ (n, A, B, C);
   - Nếu n=1 thì
     - Out: "\nChuyen “+A+" qua “+C;
   - Ngược lại
     - Push 1 bộ (n-1, B, A, C) vào s
     - Push 1 bộ (1, A, ‘ ‘, C) vào s
     - Push 1 bộ (n-1, A, C, B) vào s
4. Cho đến khi s rỗng

Tháp Hanoi không đệ qui

```
class StackA {
        private int maxSize;                // size of stack array
        private item[] stackArray;
        private int top;                    // top of stack
        //------------------------- constructor --------------------
        public StackA(int s) {
                maxSize = s;                        // set array size
                stackArray  = new item[s];   // create array
                top = -1;                           // no items yet
        }
```

Tháp Hanoi không đệ qui

```
//------------true if stack is empty ---------------
public boolean isEmpty()  {
        return (top == -1);
}
//-------------put item on top of stack -----------
public void push(item x) {
        stackArray[++top] = x ;
}
//-------------take item from top of stack --------
public item pop() {
        return stackArray[top--];
}
}
```

Tháp Hanoi không đệ qui

Khai báo class thể hiện bộ item gồm (sd, a, b, c)

```
class item {
        public int sd;
        public char ca,cb , cc;
        public void setitem(int n, char a, char b, char c)
        {
                sd = n;
                ca = a;
                cb = b;
                cc = c;
        }
}
```

Tháp Hanoi không đệ qui

```
public class ThapHNKDQ {
   public static void main(String[] args) {
        StackA  s;
        s = new StackA(200);
        item x, y;
        x = new item();
x.setitem(3, 'a', 'b', 'c');
        s.push(x);
        do {
            x = s.pop();
            if (x.sd==1)
              System.out.print("\nChuyen tu “
                          + x.ca + " qua "+x.cc);
            else  {
                y = new item();
                y.setitem(x.sd-1, x.cb, x.ca, x.cc);
                s.push(y);
                y = new item();
                y.setitem(1, x.ca, ' ', x.cc);
                s.push(y);
                y = new item();
                y.setitem(x.sd-1, x.ca, x.cc, x.cb);
                s.push(y);
             }
        } while (!s.isEmpty());
    }                          //end of main()
}
```