Organización de Computadoras

SEMANA 2

UNIVERSIDAD NACIONAL DE QUILMES

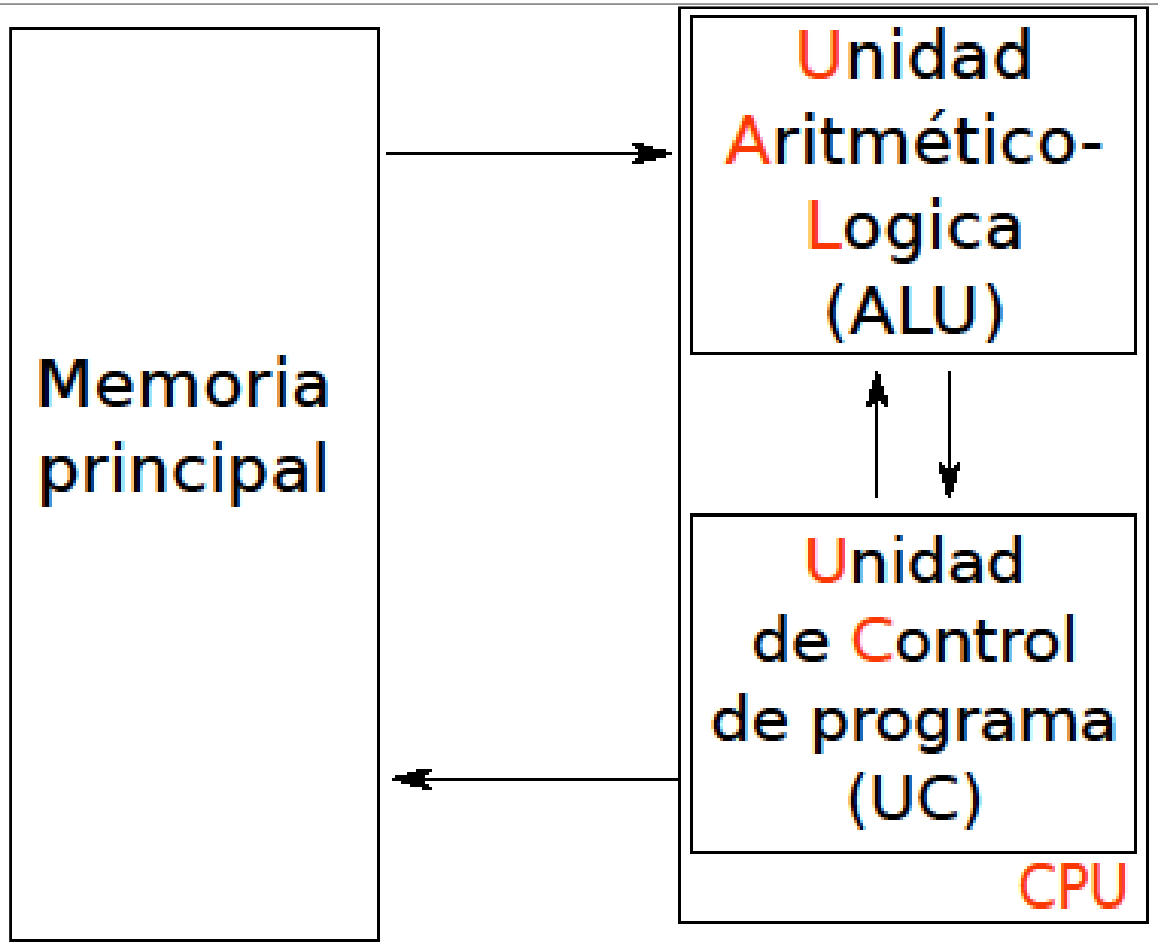
¿Qué vimos?

- Lógica proposicional ¿Para qué?
- Compuertas
  - ¿Qué son?
  - Las elementales: AND, OR, NOT
  - Algunas derivadas: NAND, NOR, XOR
- Circuitos
  - ¿Qué son?
  - ¿Cómo se construyen?
    - Enunciado
    - Tabla de verdad
    - SOP/POS
    - Circuito
- Circuitos comunes
- Circuitos Aritméticos

Hoy!

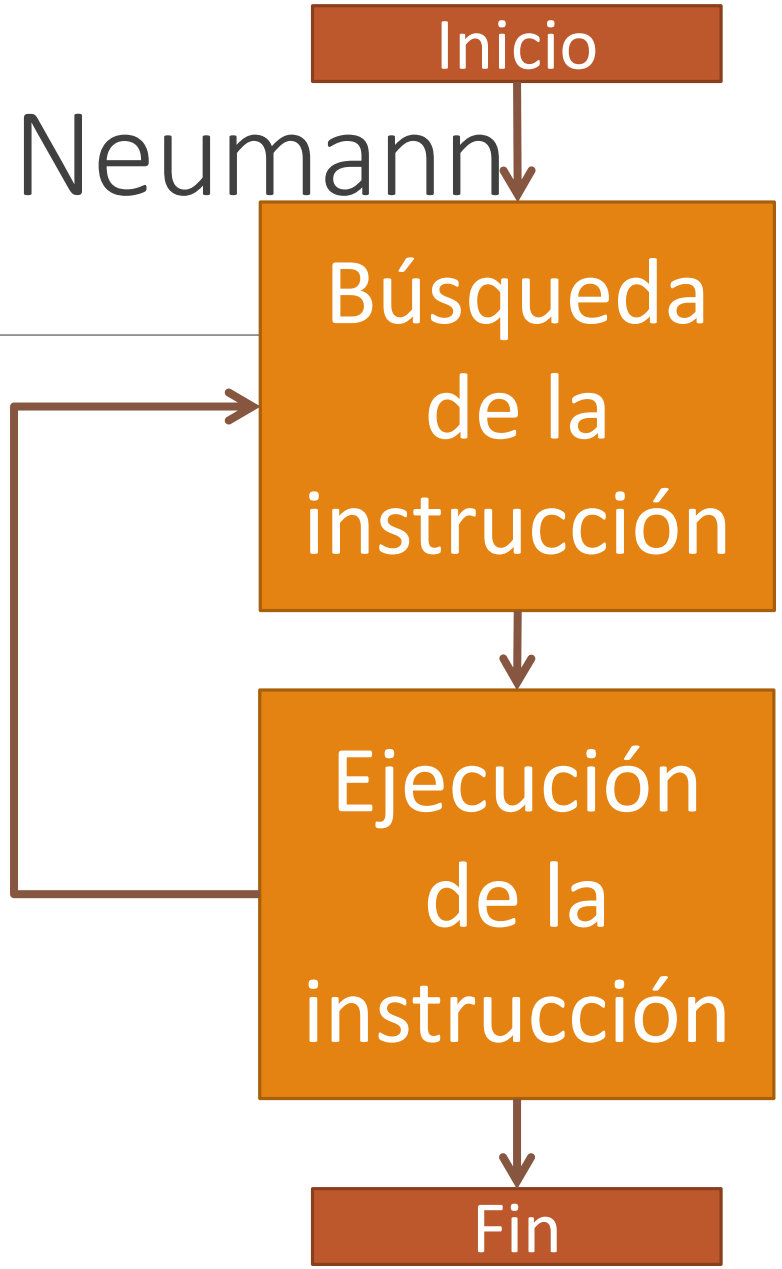
- Repaso arquitectura von Neumann
- Ensamblar y desensamblar
- Ciclo de ejecución de instrucción
- Formato de las instrucciones
- Primera máquina de uso general Q1

Arquitectura von Neumann



Arquitectura von Neumann

Ciclo de Instrucción



Arquitectura von Neumann
Unidad aritmético lógica

ALU (Unidad aritmético lógica) Dispositivo que realiza las operaciones aritméticas y lógicas sobre los datos de entrada que se le proveen

¿Cómo se usa?

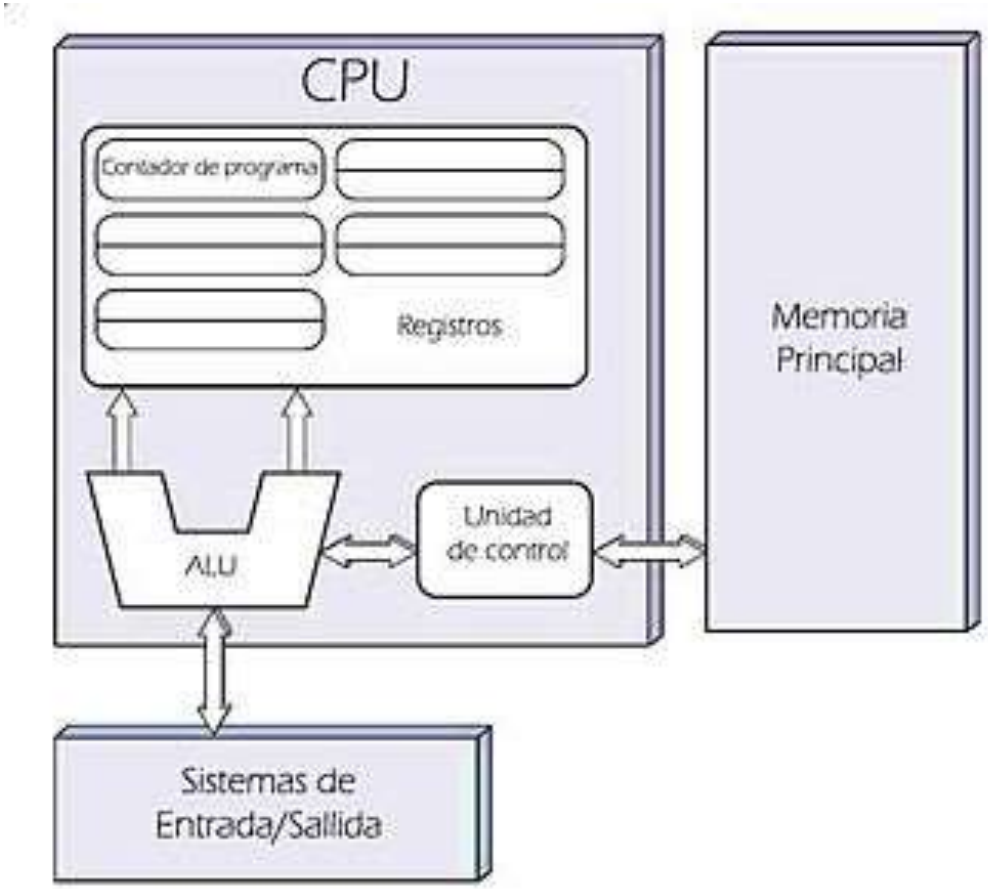
- La UC (unidad de control) suministra los operandos a la ALU
- La UC indica a la ALU la operación a llevar a cabo.
- La ALU realiza la operación.
- La UC toma el resultado.

Arquitectura von Neumann
Registros internos

Registro: Espacio de almacenamiento interno a la CPU. Es la tecnología de almacenamiento mas rápida del sistema de cómputos.

- Para ser procesado, todo dato debe alojarse en un *registro interno* a la CPU
- *Algunos* están disponibles para ser usados por los programas.
- Otros están *reservados* para uso interno de la Unidad de Control

Arquitectura von Neumann



Arquitectura von Neumann

¿Cómo se almacenan las instrucciones?

Código Binario

Código binario
Ejemplo

Código binario
Ejemplo

- Secuencia de pasos en orden






Código binario
Formato de instrucción

000

001

010

110

Código binario

Formato de instrucci

001

000

110

000

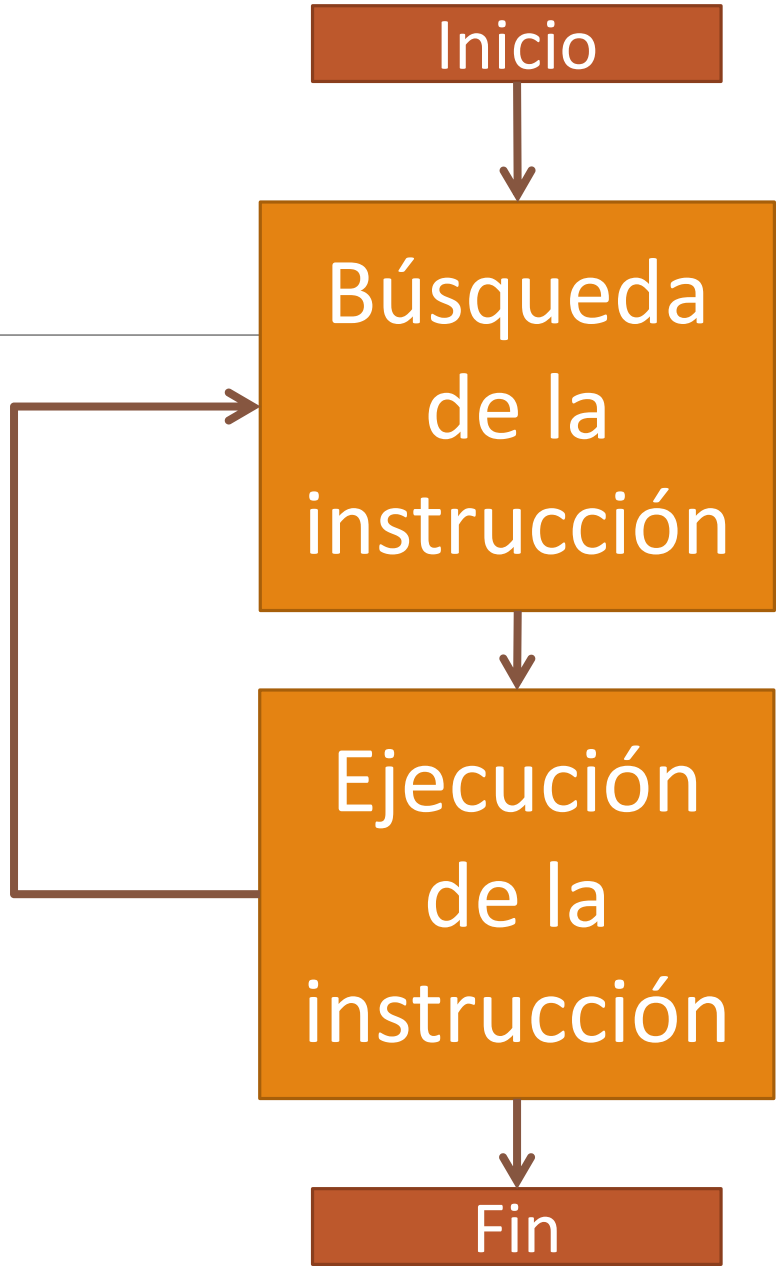
010

000

Código binario

Formato de instrucción

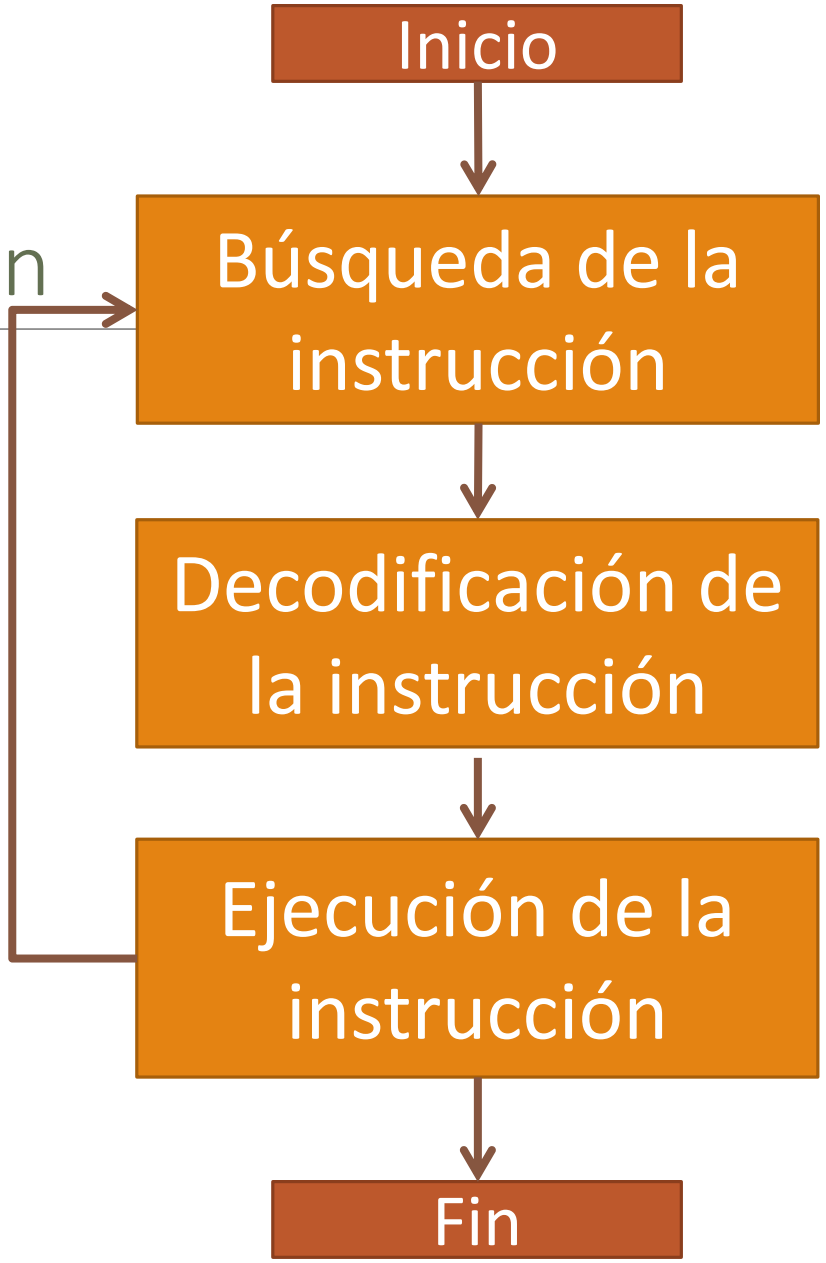
Ciclo de Instrucción



Código binario

Formato de instrucción

Ciclo de Instrucción



Código binario

- **Código Máquina**
  Código directamente interpretable por la CPU

- **Código Fuente**
  Código comprensible por un humano

Código binario

- Cuando se **desensambla**
  código máquina → código fuente

- Cuando se **ensambla**
  código fuente → código máquina

Código binario
Ejercicio




000 010 001 110

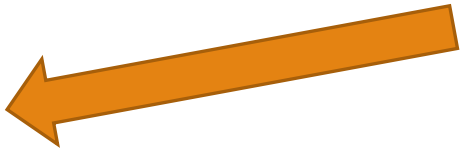
- Desensamblar:
  - 000 010 001 110 110 001
  - 000 111 001

Ciclo de vida de un programa

- El **programador** escribe el programa

Ciclo de vida de un programa

- El ensamblador lo traduce a **código máquina** y lo carga en memoria



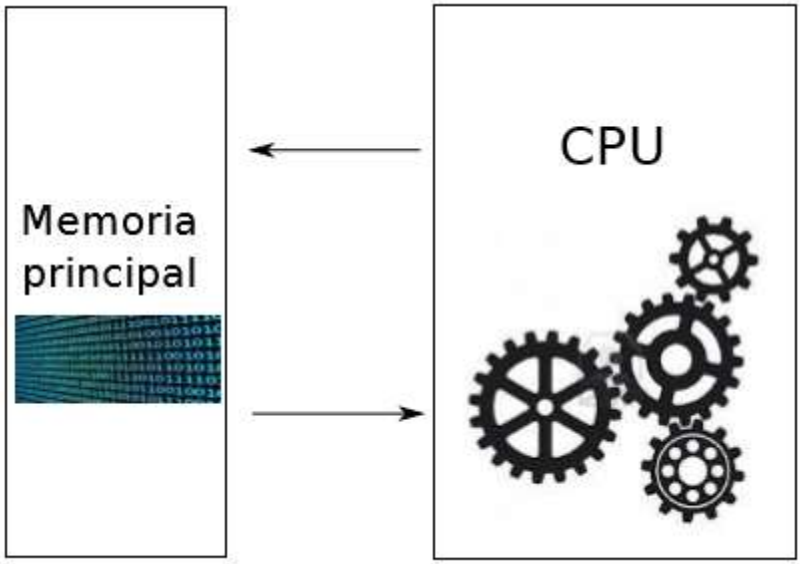
Ensamblador

Ciclo de vida de un programa

- El **usuario** pide la ejecución del programa

Ciclo de vida de un programa

- La CPU ejecuta el programa



Arquitectura

Arquitectura
Q1

- Tiene 8 registros de uso general de 16 bits: R0..R7
- Tiene instrucciones de 2 operandos:

Operación	Código	Efecto
MUL	0000	Dest ←Dest * Origen
MOV	0001	Dest← Origen
ADD	0010	Dest ← Dest + Origen
SUB	0011	Dest← Dest - Origen
DIV	0111	Dest← Dest% Origen

- Los operandos pueden ser variables (alguno de los registros) o constantes.

Arquitectura
Q1

- Ejemplos
  - MOV R0, R1
  - MUL R0, 7
  - ADD R5, 12

Arquitectura
Q1 - Modos de direccionamiento

- **Modo de direccionamiento**
  Mecanismo por el cual la unidad de control obtiene el operando requerido

- Q1 permite 2 modos de direccionamiento:
  - *modo registro*: el valor buscado está en un registro
  - *modo inmediato*: el valor buscado está codificado dentro de la instrucción (constante)

Arquitectura
Q1 - Modos de direccionamiento

- **Modo de direccionamiento Inmediato**

- **MOV R0, 0x3456**
- **ADD R6, 0xFEFE**

Arquitectura
Q1 - Modos de direccionamiento

- Modo de direccionamiento Inmediato
  - **MOV R0, 0x3456**
  - **ADD R6, 0xFEFE**

Arquitectura
Q1 - Modos de direccionamiento

- Modo de direccionamiento Registro
  - **MOV R0, 0x3456**
  - **ADD R6, R1**

Arquitectura
Q1 - Modos de direccionamiento

- ¿Tiene sentido esta instrucción?

MOV 25, R1

NO!!!

Arquitectura
Q1 - Modos de direccionamiento

Modo	Código
Inmediato	000000
Registro	100RRR

Arquitectura
Q1 – Formato de Instrucción

Arquitectura
Q1 – Ejercicios

- Ensamblemos: **MOV R1, 3**

Cod Op (4bits)	Modo Destino (6 bits)	Modo origen (6 bits)	Origen (16 bits)

Operación	Cod Op	Efecto
MOV	0001	Dest← Origen

Modo	Código
Inmediato	000000
Registro	100RRR

Arquitectura
Q1 – Ejercicios

- Ensamblemos: **ADD R1, R6**

Operación	Cod Op	Efecto
ADD	0010	Dest ← Dest + Origen

Modo	Código
Inmediato	000000
Registro	100RRR

Arquitectura
Q1 – Ejercicios

- Ensamblemos: **ADD R1, R6**

Operación	Cod Op	Efecto
ADD	0010	Dest ← Dest + Origen

Modo	Código
Inmediato	000000
Registro	100RRR

Arquitectura
Q1 – Ejercicios

- Hacer un programa que multiplique por 12 el valor de R0
- Hacer un programa que sume R0 con R1 y guarde el resultado en R2
- Hacer un programa que a R5 le reste 2 veces el valor que tiene R6
- Hacer un programa que a R4 le sume los valores de R1, R2 y R3; y le reste los valores de R5,R6 y R7

Arquitectura
Q1 – Ejercicios

- Ensamblar el siguiente programa:
  - SUB R0, R1
  - ADD R2, R0
  - DIV R2, 7
  - MUL R5, 14

Arquitectura
Q1 – Ejercicios

Cod Op (4bits)	Modo Destino (6 bits)	Modo origen (6 bits)	Origen (16 bits)

Operación	Código	Efecto
MUL	0000	Dest ←Dest * Origen
MOV	0001	Dest← Origen
ADD	0010	Dest ← Dest + Origen
SUB	0011	Dest← Dest - Origen
DIV	0111	Dest← Dest% Origen

Modo	Código
Inmediato	000000
Registro	100RRR

Arquitectura
Q1 – Ejercicios

- Desensamblar:
  - 0001100000100001
  - 0000100001100000
  - 0010100001000000
  - 0000000000000101

Arquitectura
Q1 – Ejercicios

Cod Op (4bits)	Modo Destino (6 bits)	Modo origen (6 bits)	Origen (16 bits)

Operación	Código	Efecto
MUL	0000	Dest ←Dest * Origen
MOV	0001	Dest← Origen
ADD	0010	Dest ← Dest + Origen
SUB	0011	Dest← Dest - Origen
DIV	0111	Dest← Dest% Origen

Modo	Código
Inmediato	000000
Registro	100RRR

0001100000100001

0000100001100000

0010100001000000

0000000000000101

¿Qué pasó hoy?

- Código maquina
- Formato de instrucción
- Modos de direccionamiento
- Arquitectura Q1