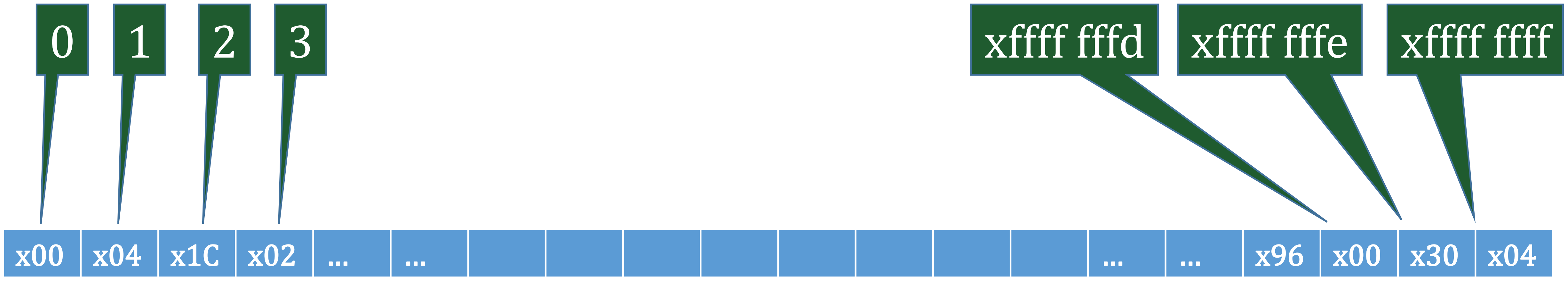
Pointers

What is a pointer?

- Says “I’m not important… what’s important is over there…
- Points AT or TO something else

Memory

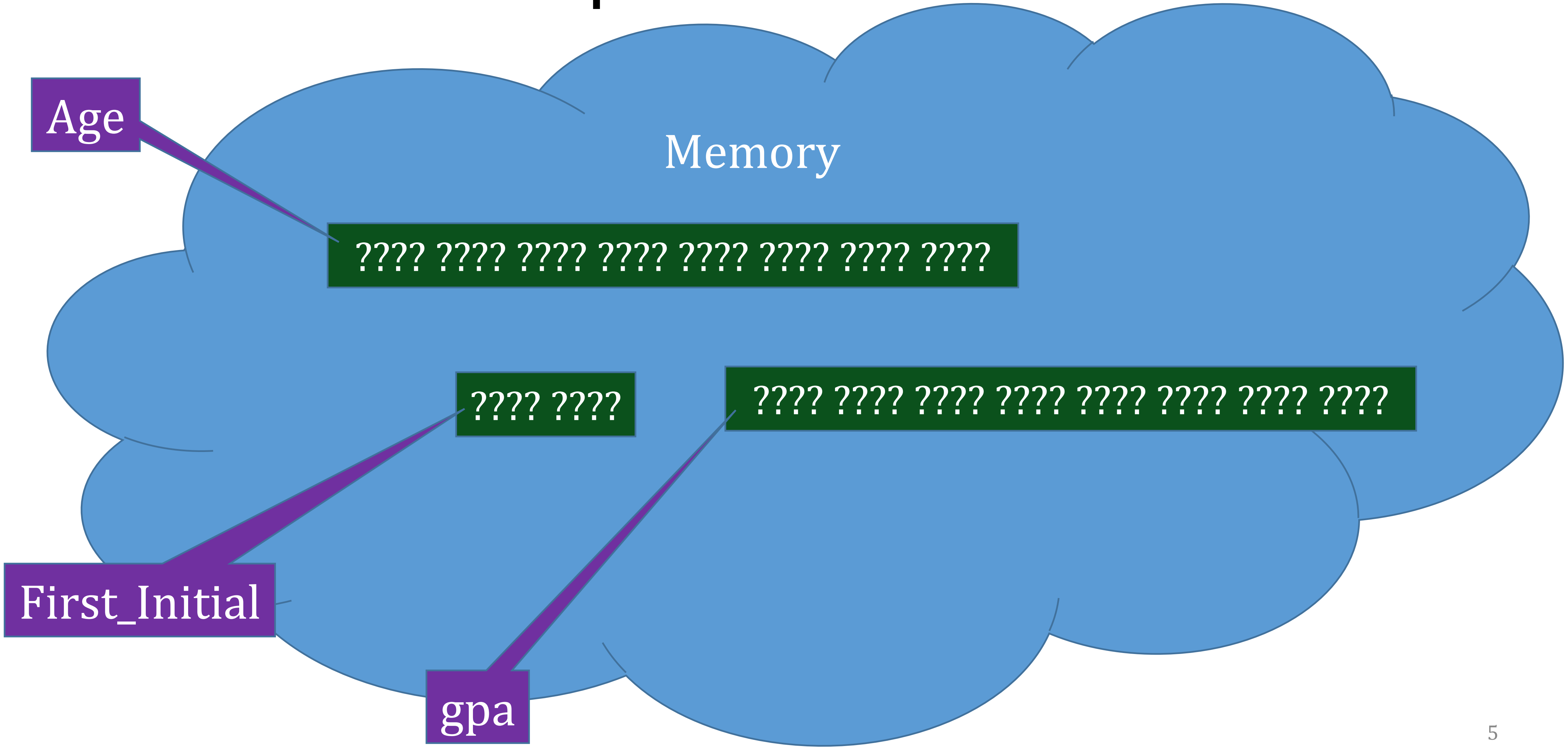
- Array of bytes
- Each element has a value



C Variables

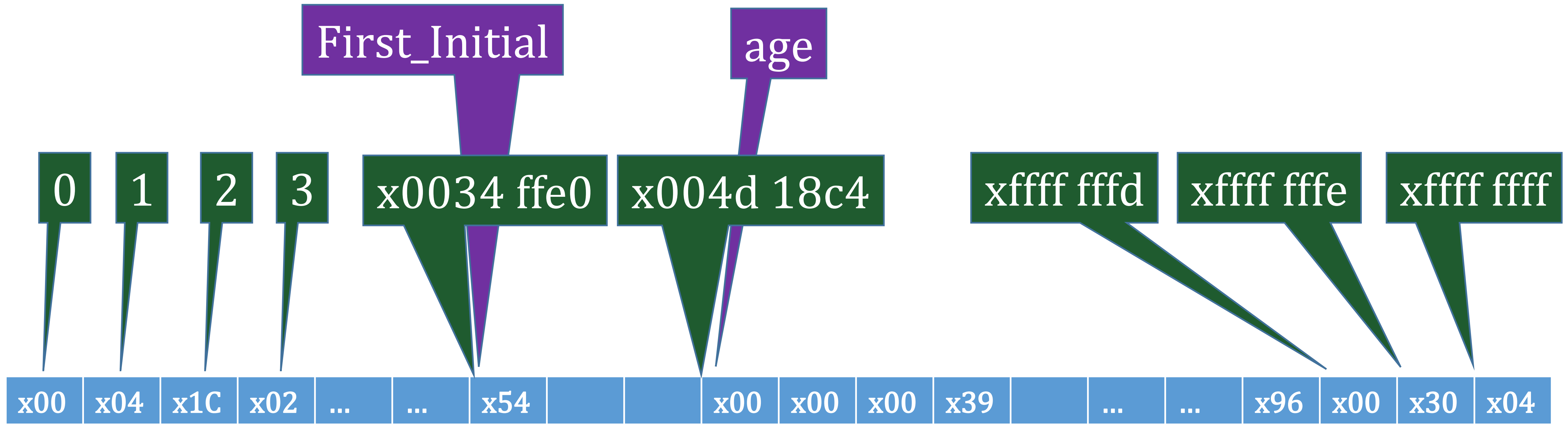
- A variable is a named piece of data
- Variables in C have…
  - A name (specified by the programmer)
  - A value (may be unassigned/unknown)
  - A location in memory (determined by the compiler)
  - A type (size and interpretation)
  - … (more to come… scope/ storage class/ etc.)
- Variables must be declared before they are used!

Variable Concept



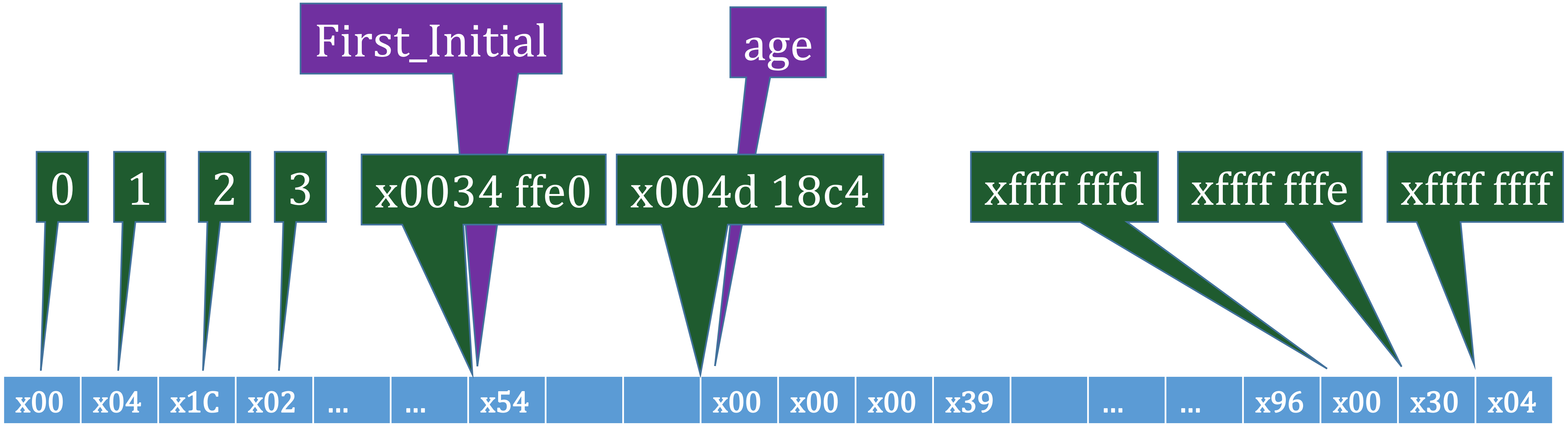
Variables In Memory

- Every variable starts at a specific location in memory
- Type of variable tells how many bytes (spaces) in memory



Variable Address/Location

- Where is the value for the variable in memory?
- The address of “First_Initial” is x0034 ffe0

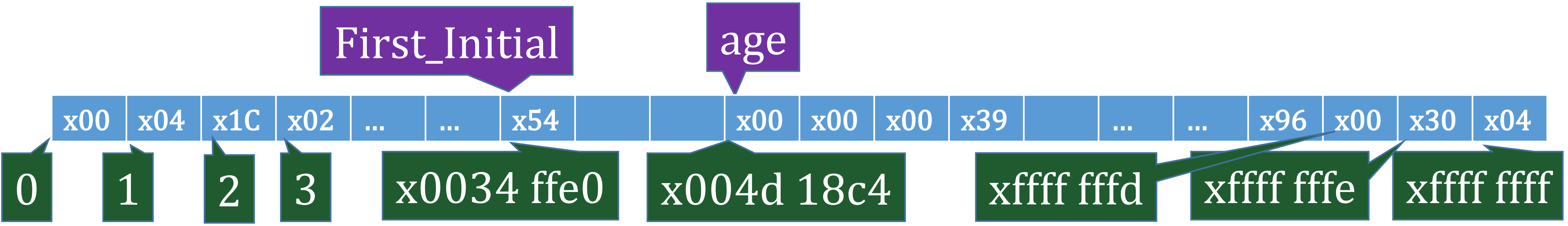


Address Of (&) operator

- An ampersand (&) in front of a variable indicates "address of"

```
char First_Initial='T';
int age=57;
printf("First_Initial is in memory at %p\n",&First_Initial);
```

```
First_Initial is in memory at 0x34ffe0
```



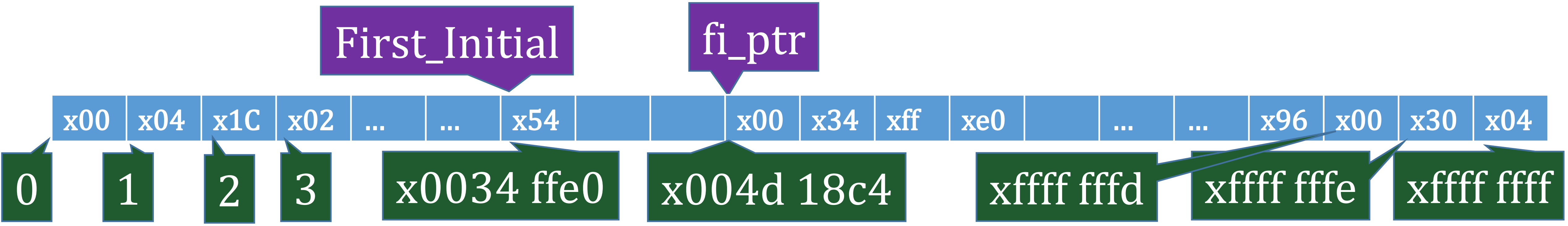
Pointers in C

- Pointers are a special data type
- The VALUE of a pointer is an address
- The TYPE of a pointer is “pointer to <target_type>
    - pointer to character
    - pointer to integer
    - pointer to float
    - pointer to array of integers
    - …

Declaring a Pointer

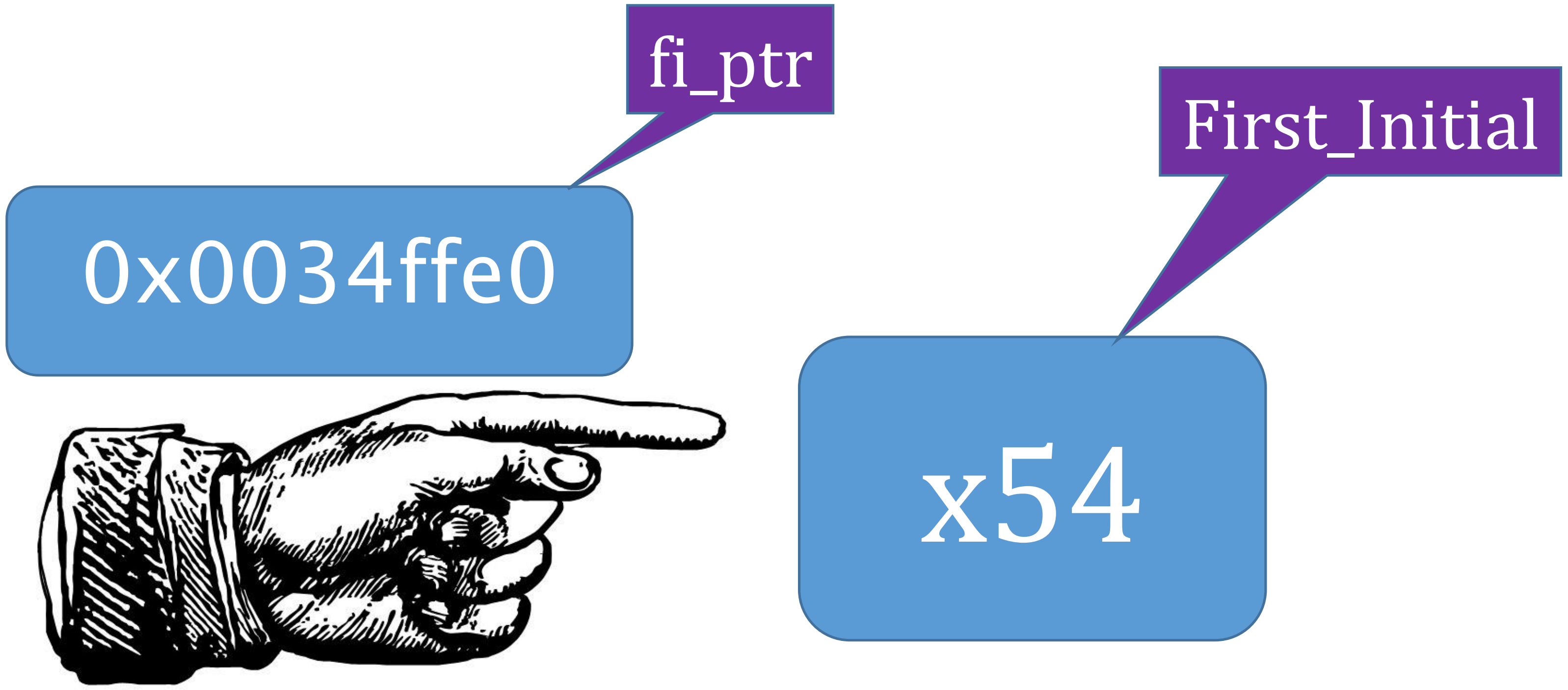
- Same as normal variable but need asterisk (*) : “pointer to”

```
char First_Initial=‘T’;
char * fi_ptr=&First_Initial;  // pointer to char
printf(“Value of fi_ptr is %p\n”,fi_ptr);
Value of fi_ptr is 0x34ffe0
```



Pointers as References

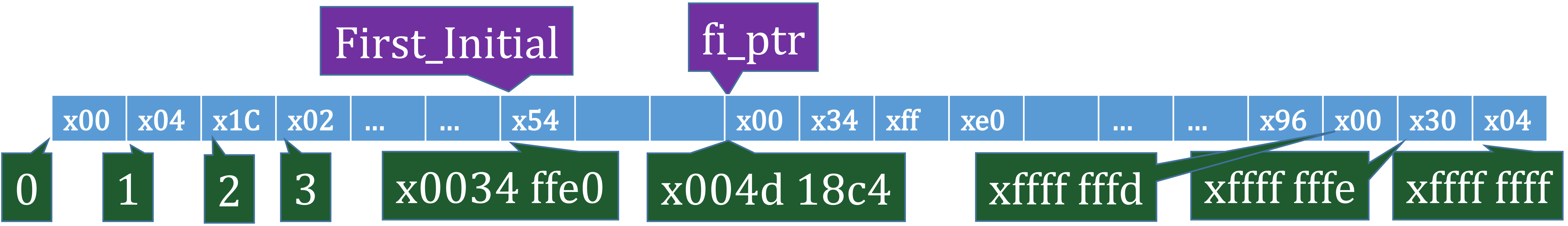
- A pointer has a value… an address in memory
- A pointer *points to* another value… the data at that address



Using a Pointer

- Same as normal variable but need asterisk (*) : “value at”

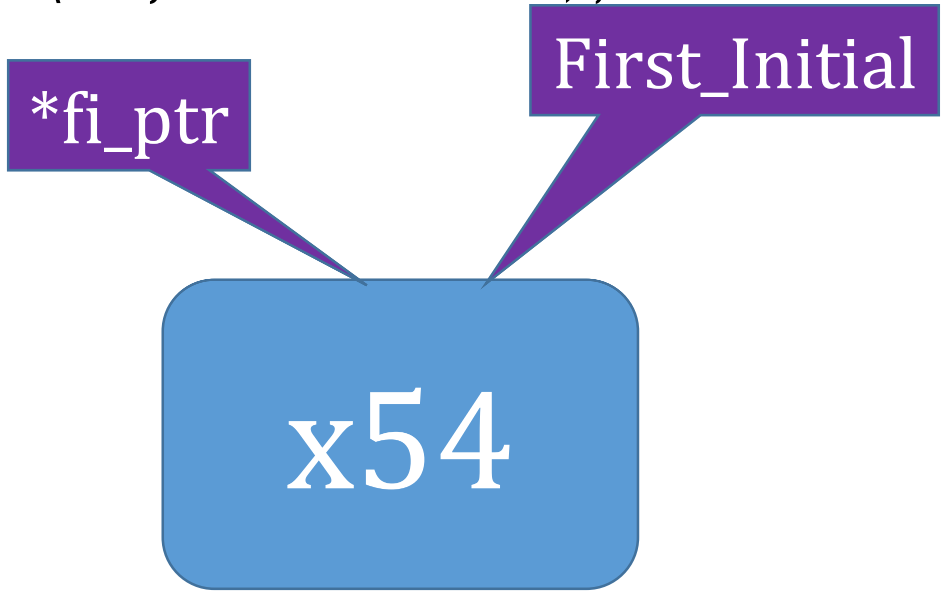
```
char First_Initial=‘T’;
char * fi_ptr=&First_Initial;   // pointer to char
printf(“fi_ptr points at %c\n”,(*fi_ptr));
fi_ptr points at T
```



Pointers as Aliases

```
char First_Initial='T';
char * fi_ptr=&First_Initial;  // pointer to char
(*fi_ptr)='A';
printf("First Initial: %c\n",First_Initial);
```

```
First Initial: A
```



Abuse of Symbols

Ampersand (&)

```
x & y // Bit-wise AND
x && y // Logical AND
&x // Address Of
```

Asterix (*)

```
x * y // multiplication
int * x // pointer to
(*x) // value at
```

Using NULL

- “NULL” is a special name whose value is 0x0000 0000.
- Beginning of Memory “belongs” to the operating system
  - General programs can read at 0, but cannot write at 0
- Therefore, we use NULL to indicate “pointer to nothing”
  - Or “pointer that we haven’t set yet”

```
int *p=NULL; // p is a pointer to nothing (for now)
…
p=&age; // Now p is a pointer to an integer
```

C Gotcha: "Dereferencing a Null Pointer"

```
int *p=NULL; // p is a pointer to nothing (for now)
int x=foo();
if (x>0) { p=&x; }
(*p) = 5;
```

Segmentation Violation when x<=0

Lab 3 Prog1- Dereference NULL

```
while (i <= argc) {
        printf("Argument %d is : %s\n",i, argv[i++]);
}
int p1=atoi(argv[1]);
```

```
$ ./prog1
Argument 0 is : prog1
Argument 1 is : (null)
Segmentation fault (core dumped)
```

Pointers have Types

- int *x; // x is a pointer to an integer
- &z – Type is: pointer to <type of z>
- (*myptr) – Type is: type which myptr is pointing to

  e.g. int *myptr=&area; (*myptr)='a';

"Unable to assign 'char' to 'int'

Void Pointers

```
void * myptr; // myptr is a pointer to void
```

- myptr is a pointer, but I'm not going to tell you what it points at
- Before you use myptr, you must cast it as a pointer to something

```
printf("myptr points to %c\n",*(char *)myptr);
```

- void * used as a "universal pointer" – a pointer to any type of data
- Programmer must know what type of data it's pointing at

Resources

- Programming in C, Chapter 10
- Wikepedia Pointers : https://en.wikipedia.org/wiki/Pointer_(computer_programming)
- C Pointer Tutorial : http://www.tutorialspoint.com/cprogramming/c_pointers.htm