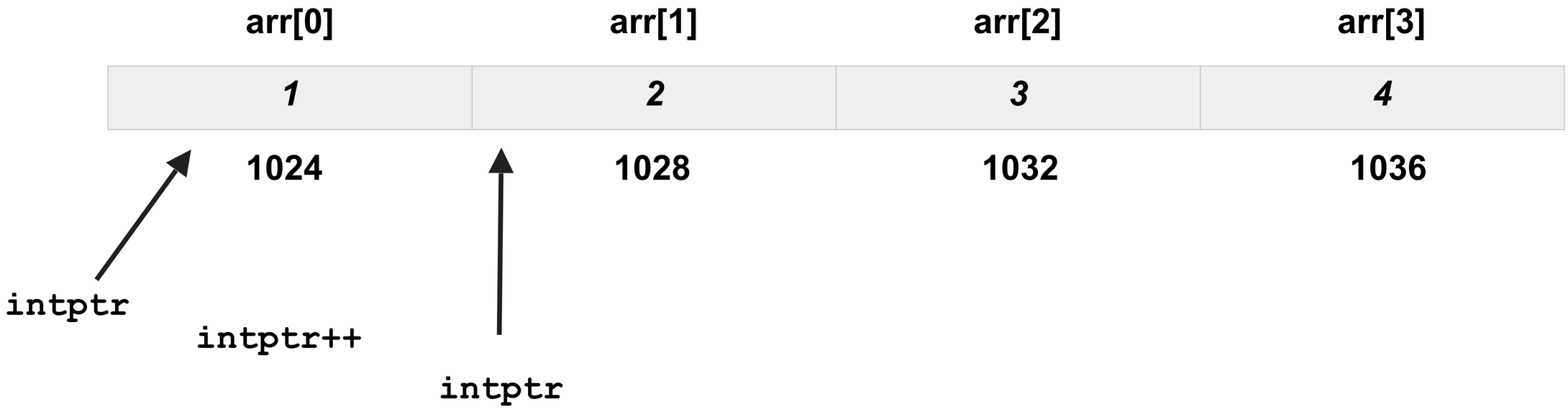
POINTER ARITHMETIC WITH ARRAYS

```
int arr[4] = { 1, 2, 3, 4 };
int* intptr = arr;
```

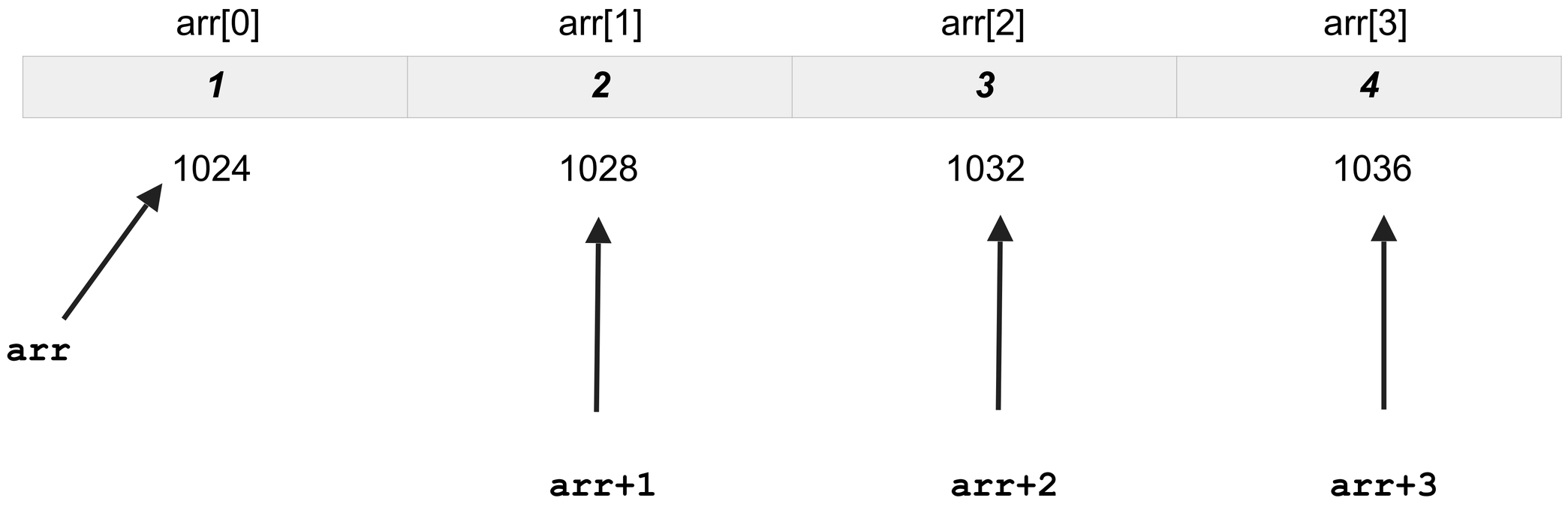


MOVES THE POINTER TO THE NEXT POSITION IN THE ARRAY I.E. IT POINTS TO `arr[1]`

`intptr` WILL NOW HAVE THE ADDRESS 1028 AND `*intptr` WILL GIVE US THE VALUE 2

INDEXING ARRAY ELEMENTS USING POINTER ARITHMETIC

```
int arr[4] = { 1, 2, 3, 4 };
```



`*(arr+1)` WILL GIVE US THE VALUE 2, IT DEREFERENCES THE ADDRESS AT `(arr + 1)` WHICH IS THE ELEMENT WE ACCESS AT `arr[1]`

Arrays in C are always laid out in contiguous portions of memory, one element after another

If you know
- the address of the very first element of the array
- how much space each element occupies

you can calculate the addresses of the remaining elements of the array

Here is where the pointer type is useful, C knows how much memory each type occupies

The programmer can work purely in terms of elements, she does not need to focus on how much memory each element occupies, C abstracts away that detail.

```
int arr[4] = { 1, 2, 3, 4 };
```

`arr + 1` will add 4 bytes to the memory location specified by arr when an integer occupies 4 bytes of memory

```
char chararray[4] = { ‘a’, ‘b’, ‘c’, ‘d’ };
```

`chararr + 1` will add 1 bytes to the memory location specified by arr as a character occupies 1 byte of memory
Both will have the address of the element at index 1

i.e. pointer arithmetic takes into account how much space is occupied by the data type the pointer points to

STRINGS IN C ARE JUST CHARACTER POINTERS

A STRING IN C IS JUST A `char*` - AN ARRAY OF CHARACTERS

```
char* str = “Hello World”;
```

IN MEMORY A STRING LOOKS EXACTLY LIKE A CHARACTER ARRAY TERMINATED BY `‘\0’`

str[0]	str[1]	str[2]	str[3]	str[4]	str[5]	str[6]	str[7]	str[8]	str[9]	str[10]	str[11]
H	***e***	***l***	***l***	***o***		***W***	***o***	***r***	***l***	***d***	***\0***
1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035

```
printf("%s", str);
```

This prints a string to screen

`*str` accesses the first element of a string which is the character ‘H’ in our example

Successive elements can be accessed using `*(str + 1), *(str + 2)` etc, they can also be accessed using `str[1], str[2]` etc

Strings in C are just character arrays