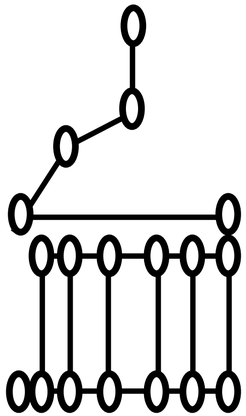
# Lecture: Pointer Arithmetic

ENGR 2730 Computers in Engineering



# Pointer arithmetic basics: what is allowed

- post increment/decrement `p++` `p--`
- pre increment/decrement `++p` `--p`
- adding or subtracting an integer `p += 5` `p - 10`
- subtracting one pointer from another `p1 - p2`

*Pointer arithmetic typically only makes sense in the context of arrays.*

# Pointer arithmetic basics: what is allowed

- increment/decrement `p++` `p--`
- pre increment/decrement `++p` `--p`
- adding or subtracting an integer `p += 5` `p - 10`
- subtracting one pointer from another `p1 - p2`

What does adding an integer to a pointer mean?

# Pointer arithmetic basics: adding an integer to a pointer

Example:

```
int a[] = {1,2,3};
int *p1;
int *p2;
p1 = a;
p2 = p1 + 2;
```

| | Address | Value |
|---|---|---|
| a[0] | 100 | 1 |
| a[1] | 104 | 2 |
| a[2] | 108 | 3 |
| | 160 | |
| | 170 | |

# Pointer arithmetic basics: adding an integer to a pointer

Example:

```
int a[] = {1,2,3};
int *p1;
int *p2;
p1 = a;
p2 = p1 + 2;
```

| | Address | Value |
|---|---|---|
| a[0] | 100 | 1 |
| a[1] | 104 | 2 |
| a[2] | 108 | 3 |
| p1 | 160 | ???? |
| | 170 | |

# Pointer arithmetic basics: adding an integer to a pointer

Example:

```
int a[] = {1,2,3};
int *p1;
int *p2;
p1 = a;
p2 = p1 + 2;
```

| | Address | Value |
|---|---|---|
| a[0] | 100 | 1 |
| a[1] | 104 | 2 |
| a[2] | 108 | 3 |
| p1 | 160 | ???? |
| p2 | 170 | ???? |

# Pointer arithmetic basics: adding an integer to a pointer

Example:

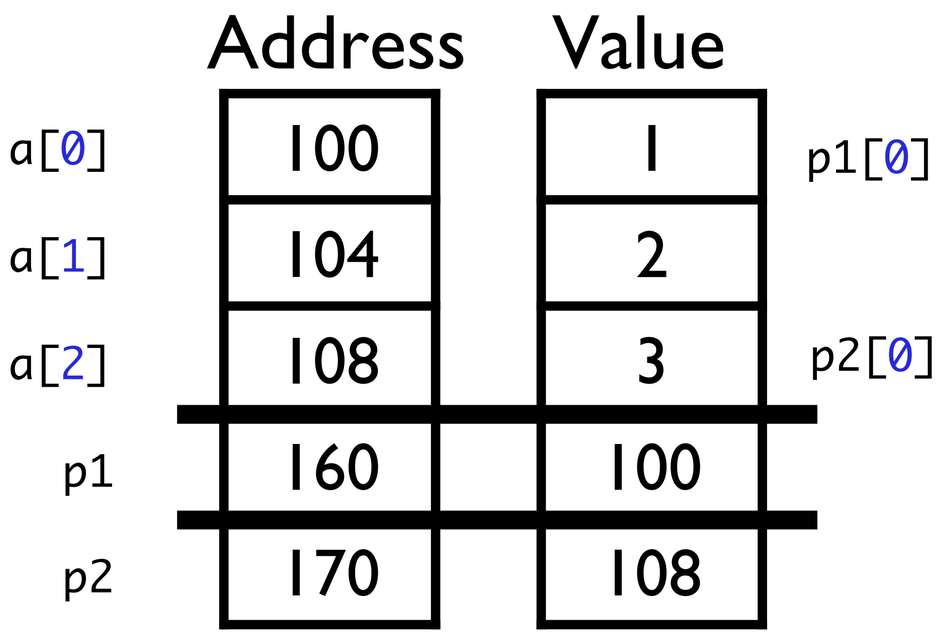
```
int a[] = {1,2,3};
int *p1;
int *p2;
p1 = a;
p2 = p1 + 2;
```

| | Address | Value | |
|---|---|---|---|
| a[0] | 100 | 1 | p1[0] |
| a[1] | 104 | 2 | |
| a[2] | 108 | 3 | |
| p1 | 160 | 100 | |
| p2 | 170 | ???? | |

# Pointer arithmetic basics: adding an integer to a pointer

## Example:

```
int a[] = {1,2,3};
int *p1;
int *p2;
p1 = a;
p2 = p1 + 2;
```




The value of p2 is 108 and not 102!
This example assumes that integers are stored in 4 bytes (i.e., 32-bits).

# Pointer arithmetic basics: adding an integer to a pointer

```
p2 = p1 + i
```

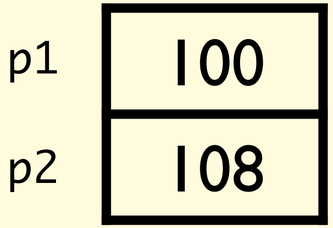
pointer to *type* integer

Suppose each variable of *type* occupies *x* bytes. Then the value of p2 will be the value of p1 plus *x* times the value of i.

*using "normal" arithmetic:*

$$p2 = p1 + x * i$$

Example:




```
int a[] = {1,2,3};
int *p1;
int *p2;
p1 = a;
p2 = p1 + 2;
```

The value of p2 is 108 and not 102!

# Pointer arithmetic basics: what is allowed

- post increment/decrement `p++` `p--`
- pre increment/decrement `++p` `--p`
- adding or subtracting an integer `p += 5` `p - 10`
- subtracting one pointer from another `p1 - p2`

What does subtracting two pointers mean?

# Pointer arithmetic basics: subtracting two pointers

Example:

```
int a[] = {1,2,3};
int *p1;
int *p2;
int i;
p1 = a;
p2 = p1 + 2;
i = p2 - p1;
```

| | Address | Value |
|---|---|---|
| a[0] | 100 | 1 |
| a[1] | 104 | 2 |
| a[2] | 108 | 3 |
| | 160 | |
| | 170 | |
| | 198 | |

# Pointer arithmetic basics: subtracting two pointers

Example:

```
int a[] = {1,2,3};
int *p1;
int *p2;
int i;
p1 = a;
p2 = p1 + 2;
i = p2 - p1;
```

| | Address | Value |
|---|---|---|
| a[0] | 100 | 1 |
| a[1] | 104 | 2 |
| a[2] | 108 | 3 |
| p1 | 160 | ???? |
| | 170 | |
| | 198 | |

# Pointer arithmetic basics: subtracting two pointers

Example:

```
int a[] = {1,2,3};
int *p1;
int *p2;
int i;
p1 = a;
p2 = p1 + 2;
i = p2 - p1;
```

| | Address | Value |
|---|---|---|
| a[0] | 100 | 1 |
| a[1] | 104 | 2 |
| a[2] | 108 | 3 |
| p1 | 160 | ???? |
| p2 | 170 | ???? |
| | 198 | |

# Pointer arithmetic basics: subtracting two pointers

Example:

```
int a[] = {1,2,3};
int *p1;
int *p2;
int i;
p1 = a;
p2 = p1 + 2;
i = p2 - p1;
```

| | Address | Value |
|---|---|---|
| a[0] | 100 | 1 |
| a[1] | 104 | 2 |
| a[2] | 108 | 3 |
| p1 | 160 | ???? |
| p2 | 170 | ???? |
| i | 198 | ???? |

# Pointer arithmetic basics: subtracting two pointers

Example:

```
int a[] = {1,2,3};
int *p1;
int *p2;
int i;
p1 = a;
p2 = p1 + 2;
i = p2 - p1;
```

| | Address | Value | |
|---|---|---|---|
| a[0] | 100 | 1 | p1[0] |
| a[1] | 104 | 2 | |
| a[2] | 108 | 3 | |
| p1 | 160 | 100 | |
| p2 | 170 | ???? | |
| i | 198 | ???? | |

# Pointer arithmetic basics: subtracting two pointers

Example:

```
int a[] = {1,2,3};
int *p1;
int *p2;
int i;
p1 = a;
p2 = p1 + 2;
i = p2 - p1;
```

| | Address | Value | |
|---|---|---|---|
| a[0] | 100 | 1 | p1[0] |
| a[1] | 104 | 2 | |
| a[2] | 108 | 3 | p2[0] |
| p1 | 160 | 100 | |
| p2 | 170 | 108 | |
| i | 198 | ???? | |

# Pointer arithmetic basics: subtracting two pointers

Example:

```
int a[] = {1,2,3};
int *p1;
int *p2;
int i;
p1 = a;
p2 = p1 + 2;
i = p2 - p1;
```

| | Address | Value | |
|---|---|---|---|
| a[0] | 100 | 1 | p1[0] |
| a[1] | 104 | 2 | |
| a[2] | 108 | 3 | p2[0] |
| p1 | 160 | 100 | |
| p2 | 170 | 108 | |
| i | 198 | 2 | |

The value of i is 2 and not 8!
This example assumes that the size of an integer is 4 bytes.

# Pointer arithmetic basics: subtracting two pointers

```
i = p2 - p1
```

integer  pointer to *type*

Suppose each variable of *type* occupies *x* bytes. Then the value of i will be the (value of p2 minus the value of p1), all divided by *x*.

*using "normal" arithmetic:*

$$i = (p2 - p1)/x$$

## Example:

| | |
|---|---|
| p1 | 100 |
| p2 | 108 |
| i | 2 |

```
int a[] = {1,2,3};
int *p1;
int *p2;
int i;
p1 = a;
p2 = p1 + 2;
i = p2 - p1;
```

The value of i is 2 and not 8!

# Practice problems

# What is printed as a result of running the following function?

```
int main()
{

    int a[3] = {2, 2, 2};
    int *ptr = nullptr;

    ptr = a;
    ptr++;

    *ptr = *ptr + 2;

    cout << a[0] << " " << a[1] << " " << a[2] << endl;

    return 0;
}
```

A. 2 4 2

B. 5 2 2

C. 3 2 2

Pick the best two lines to be added to the following code so that: (a) 2 will be added to each element of the array and (b) the values of h[3] and h[4] will be swapped.

```
int main()
{
    int h[5] = {9, 10, 11, 12, 13};
    int n = 5;

    return 0;
}

void addTwoToAllElements(int *a, int n)
{
    for (int i=0; i < n; i++)
    {
        a[i] += 2;
    }
}

void swap(int *a, int *b)
{
    int tmp = *a;
    *a = *b;
    *b = tmp;
}
```

A.
```
addTwoToAllElements(&h, 5);
swap(&(h[3]), &(h[4]));
```

B.
```
addTwoToAllElements(h, 5);
swap(&(h[3]), &(h[4]));
```

C.
```
addTwoToAllElements(&h, 5);
swap(h[3], h[4]);
```

D.
```
addTwoToAllElements(h, 5);
swap(h[3], h[4]);
```

# Select the correct line of code to be added so that 2 2 2 0 0 0 will be printed

```
int main()
{
    int arr[6] = {2, 2, 2, 2, 2, 2};

    [                                    ]

    for (int i=0; i < 6; i++)
    {
        cout << arr[i] << " ";
    }
    cout << endl;

    return 0;
}
```

A. `setAllToZero(&arr, 3);`

B. `setAllToZero(arr, 3);`

C. `setAllToZero(&arr[3], 3);`

D. `setAllToZero(arr[3], 3);`

E. `setAllToZero(&arr, 6);`

F. `setAllToZero(arr, 6);`

G. `setAllToZero(&arr[3], 6);`

H. `setAllToZero(arr[3], 6);`