Today's plan

- Class interactions
  - Group discussion on Ex 4-2
  - Recap the concepts
  - Quiz
- Class exercises
  - Ex 4-3

Ex 4-2

Group discussion

Share your implementation with your group and discuss these questions:

- What memory problem do you find in `pairwise_sum.c` and how do you fix it?
- What memory problem do you find in `prime.c` and how do you fix it?
- What is the takeaway from this exercise?
  - You must `free` what you have allocated. Beware of losing your pointer by nested function calls as in `pairwise_sum.c`.
  - Beware of what you need to pass in the function. When using pointer-to-pointer type, the compiler will not give you a warning even if you are passing the wrong address. The reason is, it is just a pointer type to a type. A pointer type to "an integer pointer" and a pointer type to "an integer" are both addresses.

Recap - pointer arithmetic

- `ptr1 = ptr2;` same as value assignment, but it is copying address stored in `ptr2` to `ptr1`. Not the actual values in the memory.
- Then, `ptr1` and `ptr2` have the same address, i.e. pointing to the same memory.
- If we do `*ptr1 = 100`, `*ptr2` will get 100 as well. (because it is the same memory.)
- If we treat pointers as integer values (they are addresses), the assignment and comparison are the same logic as integer assignment and comparison.
- However, the arithmetic is different (for addition and subtraction)!
- If we do `ptr + a`, it will advance the address by `a` $\times$ the size of the data type in bytes.
- e.g. `int *p;` If `p` is 220 (in decimal), then `p + 4` is 236.

Recap - pointers and arrays, multi-dimensional dynamic arrays

- Recall that: we say an array is the same as a constant pointer.
- We have `array[x]` is the same as `*(array + x)`.
- The square bracket operator of an array is the same as advancing the address by `x` then de-referencing.
- e.g. `float a[200]`. `a[4]` and `*(a + 4)` are the same.
- Create a 100×100 integer array dynamically:

```
1  int **a = malloc(sizeof(int*)*100);
2  for (int i = 0; i < 100; ++i)
3      a[i] = malloc(sizeof(int) * 100);
```

- Caution: it could be non-linear!

Recap - Reading C declarations

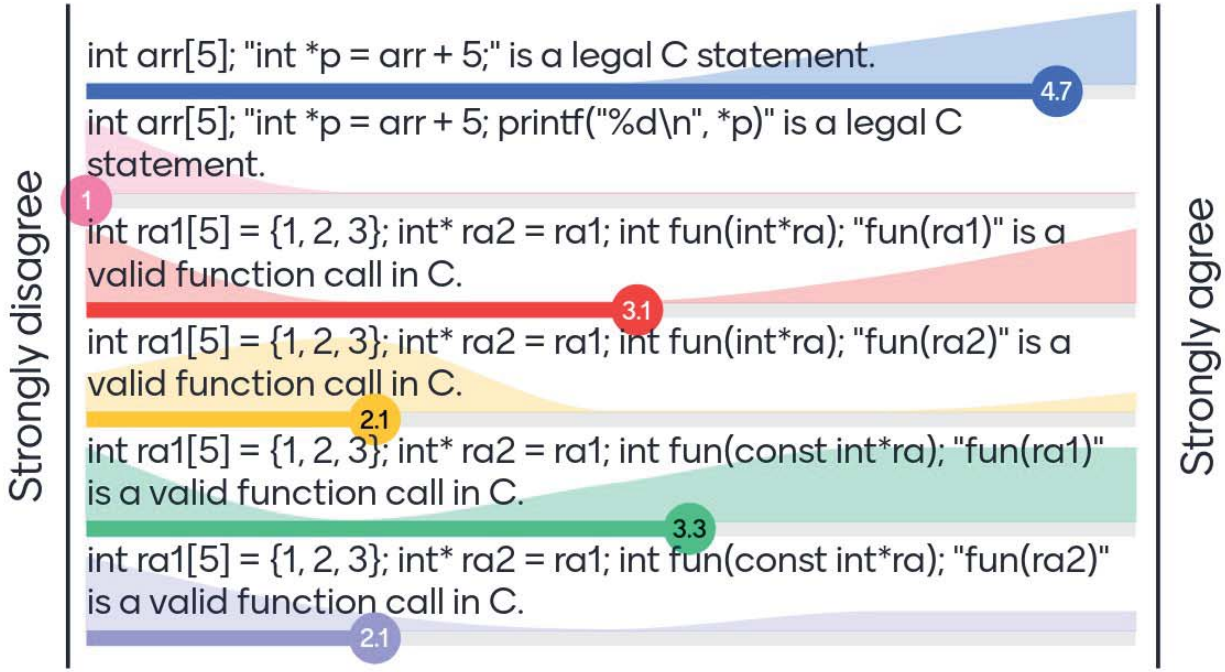
- The "right-left" rules:
  - `*`: read as "a pointer to"
  - `[]`: read as "an array of"
  - `()`: read as "a function returning"
- Staring from the variable name, right first, then left. Examples:
  - `int (*foo)[]`: read as "foo is a pointer to an array of integers"
  - `const int (*const foo)[]`: read as "foo is a constant pointer to an array of integers which are constant"
  - `int const * const (*foo)[]`: read as "foo is a pointer to an array of constant pointers to constant integers"
  - `int *bar()`: read as "bar is a function returning a pointer to an integer"
  - `int (*bar)()`: read as "bar is a pointer to a function returning an integer"
  - `int (*bar())()`: read as "bar is a function returning a pointer to a function returning an integer"

Quiz

Quiz!

What do you think about the below statements?

Mentimeter



7

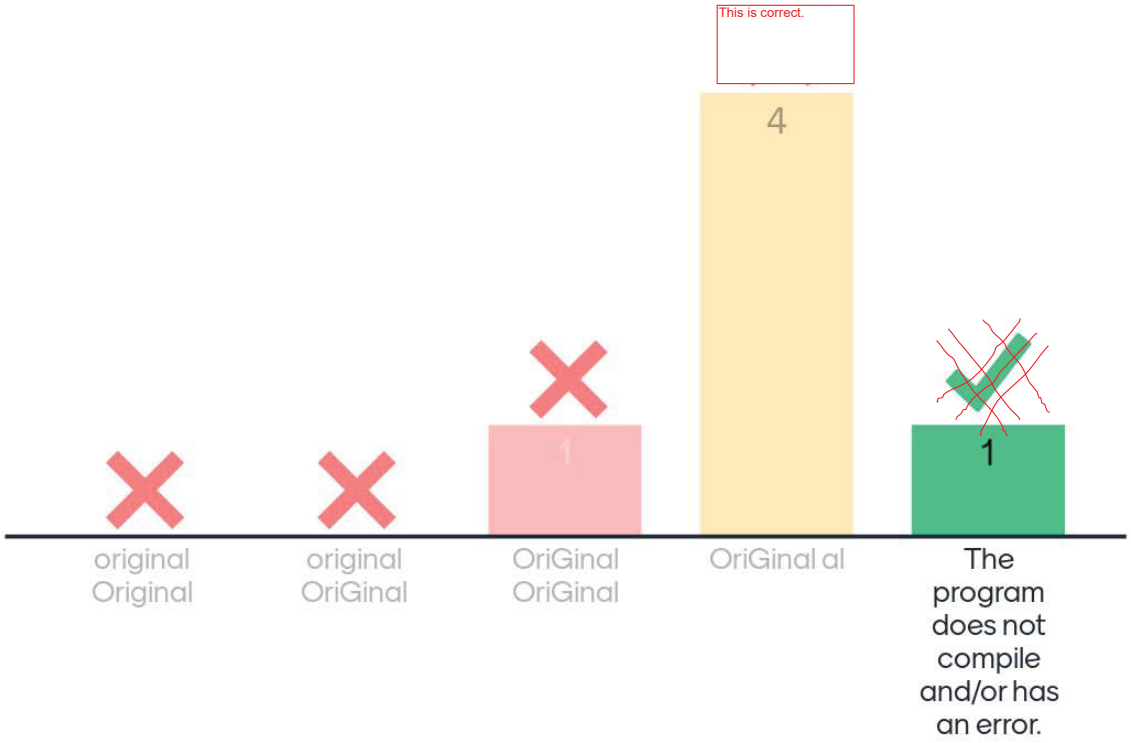
Quiz

What output is printed by the code below?

```
1  char str1[] = "original";
2  char * str2;
3
4  str2 = str1;
5  *str2 = 'O';
6  str2 += 3;
7  *str2 = 'G';
8  str2 += 3;
9
10 printf("%s %s\n", str1, str2);
```

What output is printed by the code shown on the slide?

Mentimeter



6

Quiz

What output is printed by the code below?

```
1  int arr[] = { 94, 69, 35, 72, 9 };
2  int *p = arr;
3  int *q = p + 3;
4  int *r = q - 1;
5  printf("%d %d %d\n", *p, *q, *r);
6  ptrdiff_t x = q - p;
7  ptrdiff_t y = r - p;
8  ptrdiff_t z = q - r;
9  printf("%d %d %d\n", (int)x, (int)y, (int)z);
10 ptrdiff_t m = p - q;
11 printf("%d\n", (int)m);
12 int c = (p < q);
13 int d = (q < p);
14 printf("%d %d\n", c, d);
```

What output is printed by the code shown on the slide?

Mentimeter

94 72 353 2 1 -31 0

94 35 69

94 69 35 72 9 97 71 38 75 12

94 72 35; 3 2 -1;

94 72 353 2 1 -31 0

94 72 353 2 1 -30 1

94 72 353 2 1 -31 0

94 72 353 2 1 -31 0

8

What output is printed by the code shown on the slide?

Mentimeter

94 72 35\n 3 2 1\n -3\n 1 0 ×

94 72 35 3 2 1 .... ×

94 72 35 3 2 1 -3 1 0 ×

The correct answer is: 94 72 35\n3 2 1\n-3\n1 0\n

3

Quiz

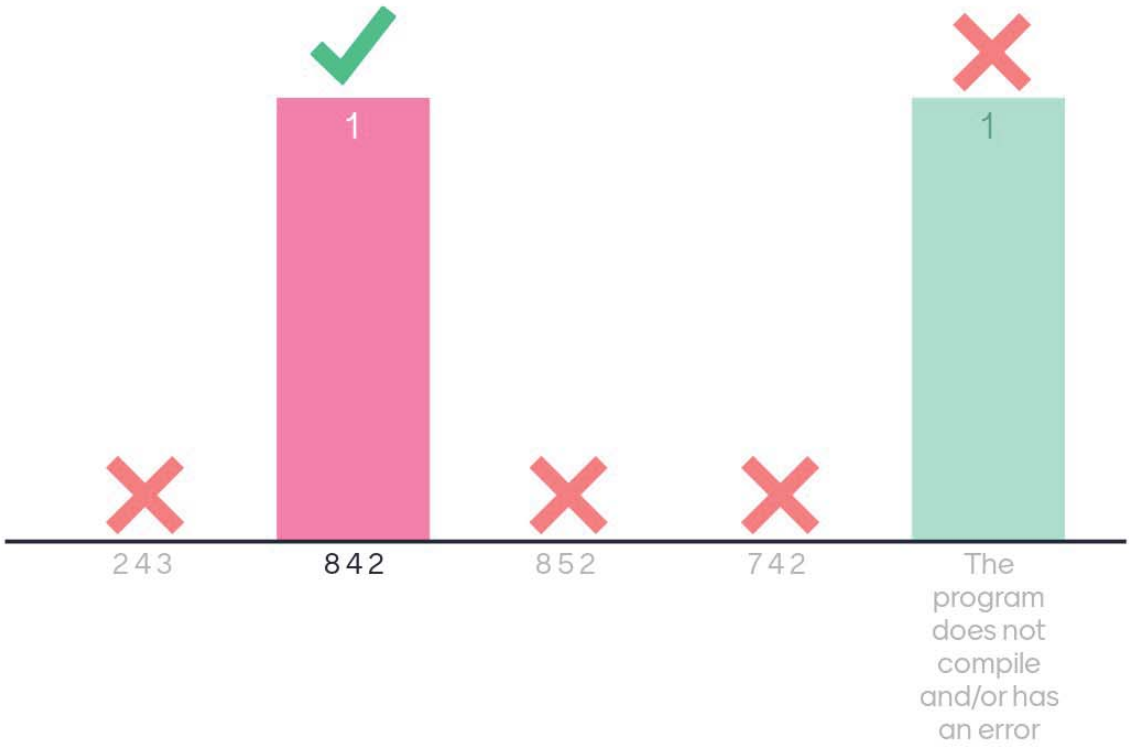
What output is printed by the code below?

```c
#include <stdio.h>

int main() {
  int a[] = {1, 1, 2, 3, 4};
  printf("%d ", (*a) + 7);
  printf("%d ", *(a + 4));
  printf("%d ", *(&a[1] + 1));
  return 0;
}
```

What output is printed by the code shown on the slide?

Mentimeter



2

Quiz

What output is printed by the code below?

```c
#include <stdio.h>
int sum(int a[], int n) {
  int x = 0;
  for (int i = 0; i < n; i++) {
    x += a[i];
  }
  return x;
}
int main(void) {
  int data[] = { 23, 59, 82, 42, 67, 89, 76, 44, 85, 81 };
  int result = sum(data + 3, 4);
  printf("result=%d\n", result);
  return 0;
}
```

What output is printed by the code shown on the slide?

Mentimeter

274 ✕ 3x

The correct answer is: Let's discuss with your classmates in a breakout room.

3

Type Answer

Mentimeter

The correct answer is: result=274

Ask me anything

Mentimeter

0 questions
0 upvotes

Class exercises

Ex 4-3