

CSC 209 Review 5 Solution

August 21, 2020

1. a) 14
b) 34
c) 4
d) true
e) false

Notes

- **Pointer Arithmetic**

- Adding an integer to a pointer

Example



- Subtracting an integer from a pointer

Example



- Subtracting one pointer from another

Example



• Comparing pointers

- Can compare pointers using relational operators (i.e. `<`, `<=`, `>`, `>=`) and the equality operators (i.e. `==`, `!=`)
- Returns 1 if **true** and 0 if **false**

Example

```
p = &a[5];
q = &a[1];
```

`p <= q` is 0 and `p >= q` is 1

2. `low` and `high` are memory addresses.

So, `low + high` is out of bound, and it could potentially point to an undesirable or wrong value.

To fix this, we subtract the from high value to the low value:

$$\text{middle} = \frac{\text{low} + \text{high}}{2} \quad (1)$$

3. I need to write the contents of an array `a` after the execution of statements outlined in problem sheet.

After execution, the array would have contents of `[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]`.

Notes

- **Combining the `*` and `++` Operators**

- `*p++` or `*p++` → Value of expression is `*p` before increment; increment `p` later
- `(*p)++` → Value of expression is `*p` before increment; increment `*p` later
- `++*p` or `*(++p)` → Increment `p` first; value of expression is `*p` after increment
- `+++p` or `++(*p)` → Increment `*p` first; value of expression is `*p` after increment

Example

`a[i++] = j`

Means assign the value `j` to `a[i]` before increment

Example 2

```
for (p = &a[0]; p < &a[N]; p++)  
    sum += *p;
```

Is the same as

```
p = &a[0];  
while (p < &a[N])  
    sum += *p++;
```

4. I need to re-write prototype `make_empty`, `is_empty` and `is_full` of the following code to use the pointer variable `top_ptr` instead of the integer variable `top`.

```
1  #include <stdbool.h>
2
3  #define STACK_SIZE 100
4
5  /*external variables*/
6  int contents[STACK_SIZE]
7  int top = 0;
8
9  void make_empty(void) {
10     top = 0;
11 }
12
13 bool is_empty(void) {
14     return top == 0;
15 }
16
17 bool is_full(void) {
18     return top == STACK_SIZE;
19 }
```

And after re-write using `top_ptr` instead of `top` have:

```
1  #include <stdbool.h>
2
3  #define STACK_SIZE 100
4
5  /*external variables*/
6  int contents[STACK_SIZE]
7  int *top_ptr = &contents[0];
8
9  void make_empty(void) {
10     top_ptr = &contents[0];
11 }
12
13 bool is_empty(void) {
14     return top_ptr == &contents[0];
15 }
16
17 bool is_full(void) {
18     return top_ptr == &contents[STACK_SIZE-1];
19 }
```

5. I need to identify which

Notes

- Using an Array Name as a Pointer

- The name of an array can be used as a pointer to the first element in the array.

Example

```
int a[10];
```

```
*a = 7; /* stores 7 in a[0] */
```