

CSC 209 Review 4 Solution

August 19, 2020

1. The answer is a) `*p` and g) `*&i`.

Notes

- **Address and Indirection Pointers**

- If `x` is a variable, `&x` points to its memory address
- `*` in `*p` is called **Indirection operator**
 - * Allows variable to gain access to the object pointed by `p`

- **Aliases**

- Is the situation where the value in same memory location can be accessed using different variable names.

Example 1:

```
int i, p*;
p = & i;
printf("%d\n", *p); /* *p is an alias of i */
```

Example 2:

```
int i, p*;
p = *&i /* *p is an alias of i */
```

2. The answers are b) `*p = &i;`, f) `p = q;`, and i) `*p = *q;`

Correct Solution

The answers are e) `p = *&q;`, f) `p = q;`, and i) `*p = *q;`

`p = *&q;` is the same as `p = q`

Notes

- The `*` operator turns a *value* of type **pointer** to **T** into a *variable* of type **T**.
- The `&` operator turns a *variable* of type **T** into a *value* of type pointer to **T**.

- **Pointer Assignment**

- The following is an example of correct pointer assignment

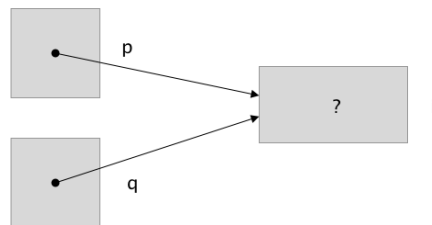
```
int i, j, *p, *q;  
p = &i;
```

* Means the memory address of **p** is pointing to memory address of **i**

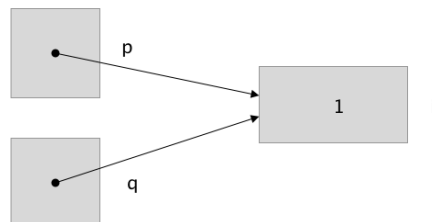
- The following is another valid example of pointer assignment

```
int i, j, *p, *q;  
p = &i;  
q = p;
```

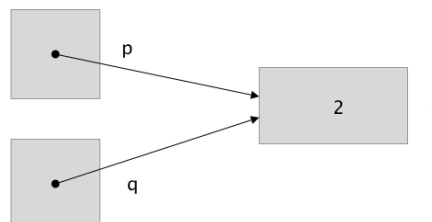
* Means memory address of **q** is the memory address of **p** (which is the memory address of **i**)



```
*p = 1;
```



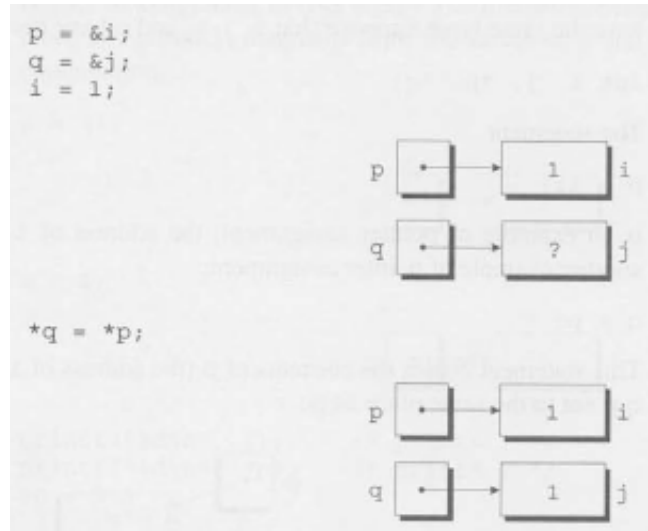
```
*p = 2;
```



- The following is not a pointer assignment

`*q = *p`

- * It copies the value that p points to



```

31 void avg_sum(double a[], int n, double *avg, double *sum)
2   {
3       int i;
4
5       *sum = 0.0;
6       for (i = 0; i < n; i++)
7           *sum += a[i];
8       *avg = *sum / n;
9   }

```

Notes:

- **Pointer as Arguments:**

- Construct prototype using pointer variable as parameter so it can be passed by reference

Example

```

void decompose(double x, long *int_part, double *frac_part);
or
void decompose(double, long *, double *);
void decompose(double x, long *int_part, double *frac_part)
{
    *int_part = (long) x;
    *frac_part = x - *int_part;
}

```

- When using the prototype, pass variable to prototype by reference using & operator (points to variable's memory location)

```
decompose(3.14159, &i, &d);
```

```
41 void swap(int *p, int *q) {  
2   int temp;  
3  
4   temp = *p;  
5   *p = *q;  
6   *q = temp;  
7 }
```

```
51 void split_time(long total_sec, int *hr, int *min, int *sec) {  
2   int hours, mins, seconds, min_sec;  
3  
4   hours = total_sec % 60;  
5   min_sec = total_sec - hours;  
6   mins = min_sec % 60;  
7   seconds = min_sec - mins;  
8  
9   *hr = hours;  
10  *min = mins;  
11  *sec = seconds;  
12 }
```