1. break causes control to exit the current loop or switch statement. exit causes control to exit the current program. Use break when you want to stop looping or executing the current case. Use exit when you want to stop executing the program.

2. A causes pointer p1 to point to the same place p2 points. B causes the space pointed to by p1 to contain a copy of the string pointed to by p2.

3. Inside a function, A creates a variable that retains its value between function invocations; outside a function, A creates a global variable that cannot be 'seen' from other files. B declares that the variable i is defined in a different file.

4. A creates an array of 12 arrays, each with space for 20 chars. B creates an array of 12 character pointers, but allocates no space for strings. Use A when you expect limited length strings, and use B when the lengths of the strings is unknown and will be determined at run time.

5. A copies from where p points to c then prints out c until a null byte is copied. B prints out the char stored in c for as many times as that character appears where p points.

6. In A, the value stored in variable n is passed to the function f. f() may operate on that value. In B, the address of the variable n is passed to the function f. f() may use that address to modify the value stored in n.

7. s will not equal NULL. At the end of the string, s will point to the value `'\0'`. Change (s!=NULL) to (`*s != '\0'`)

8. This is a logic error. Assuming MIN <= MAX, the test is true for all values. Instead of || use &&

9. newnode is a pointer, not a struct. You need to use "->" not "."

10. a) 7 b) 5 c) 2 d) 3 e) pointer to integer

11. a) local x, global y
b) local c, param y, global x
c) local d, global x and y

```
/* #12A
 * sum 24 temps, divide by 24
 */

float avg_temp( int t[][24], int d )
{
    int total=0, hour=0;
    while( hour< 24 )
     total += t[d][hour++];
    return total/24.0;
}


/* #12B
 * initialize max-so-far to first val
 * compare it to each of the rest
 * report day of largest seen
 */
int scorcher( int t[][24] )
{
   float maxtemp, f;
   int   hotday=0, d=1;

   for( maxtemp=avg_temp(t,0) ; d<7; d++ )
                              if ( (f=avg_temp(t,d)) > maxtemp ){
                                                          hotday = d ;
                                                          maxtemp = f;
                              }
   return hotday;
}

/* #12C
 * nested loop covers days and hours
 * prints day, hour for each item <= 32
 */

char *days[] = "Mon", "Tue", "Wed", "Thu",
               "Fri", "Sat", "Sun" ;

freezing( int t[][24] )
{
   int d, h;
   for( d=0;d<7;d++ )
      for( h=0;h<24;h++ )
         if( t[d][h] <= 32 )
            printf("%s %d\n", days[d], h);
}
```

```
13A

/*
 * traverse linked list, keep coun
 * return counter when item found
 * or return -1 if item not found
 */
int word_position(struct word *h,
{
    int
    struct word *p ;
    for( p=h->next ; p ; p=p->next
        if ( strcmp(p->name, s) ==
           return n;
    return -1;
}

13B

/*
 * draw a picture for this one
 */
void swap_top2( struct word *h )
{
    struct word *newfirst;

    newfirst       = h->next->next
    h->next->next  = newfirst->nex
    newfirst->next = h->next;
    h->next        = newfirst;
}
```