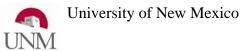


You may use one page of hand written notes (both sides) and a dictionary. No i-phones, calculators nor any other type of non-organic computer.

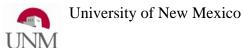
1) If Logic: This C program compiles and runs. What is its output?

```
1) #include <stdio.h>
 2) void main(void)
 3) {
 4)
      int a = 14;
 5)
      if (a % 7 == 0)
 6)
      { printf("A\n");
 7)
        a+=2;
 8)
      }
 9)
      else
10)
      { printf("B\n");
11)
        a+=4;
12)
13)
      printf("C\n");
14)
      if (a % 7 == 0)
15)
      { printf("D\n");
16)
        a+=3;
17)
      }
18)
      else
19)
      { printf("E\n");
        if (a > 5)
20)
21)
        { printf("F\n");
22)
          a +=4;
23)
24)
        else if (a > 10)
25)
        { printf("G\n");
26)
          a = 4;
27)
28)
29)
      printf("%d\n",a);
30) }
```



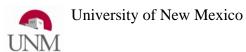
2) Variable Scope: This C program compiles and runs. What is its output?

```
1) #include <stdio.h>
 2)
 3) int b=5;
 4)
 5) int foo(int n)
 6) {
      int a=3;
 7)
 8)
      a++;
 9)
      b++;
10)
      n = n+a+b;
11)
      printf("foo: n=%d, a=%d, b=%d \n", n, a, b);
12)
      return n;
13) }
14)
15) void main(void)
16) {
17)
      int a, n;
18)
      n = 5;
19)
      a = foo(n);
20)
      printf("main: n=%d, a=%d, b=%d\n", n, a, b);
21)
22)
      a = foo(n);
23)
      printf("main: n=%d, a=%d, b=%d\n", n, a, b);
24) }
```



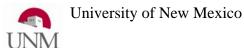
3) Binary Search: This C program compiles and runs. What is its output?

```
1) #include <stdio.h>
 2)
 3) #define ENDCODE -1
 4) int binarySearch(int x, int v[])
 5) {
 6)
      int mid;
      int low = 0;
 7)
 8)
      int high = 0;
      while (v[high] != ENDCODE) high++;
 9)
10)
      high--;
      while (low <=high)</pre>
11)
12)
      {
13)
       mid = (low+high)/2;
        printf("[%d %d %d] ", low, mid, high);
14)
15)
16)
        if (x < v[mid]) high = mid-1;
17)
        else if (x > v[mid]) low = mid+1;
18)
        else return mid;
      }
19)
20)
      return -1;
21) }
22)
23) void main(void)
24) {
      int nums[]={12, 13, 15, 17, 21, 23, 27, 39, 43, 51, -1};
25)
      printf("idx = %d\n", binarySearch(21, nums));
26)
      printf("idx = %d\n", binarySearch(30, nums));
27)
28) }
```



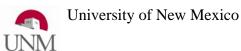
4) Bit Operators: This C program compiles and runs. What is its output?

```
1) #include <stdio.h>
 2) void main(void)
 3) { unsigned char x = 60;
 4)
 5) unsigned char a = x << 4;</p>
 6) unsigned char b = x >> 4;
     unsigned char c = x & 15;
 7)
     unsigned char d = x & 240;
 8)
 9) unsigned char e = x | 15;
10) unsigned char f = x ^ 15;
11)
12) printf("a=%d, b=%d, c=%d, d=%d, e=%d, f=%d\n",
             a, b, c, d, e, f);
13)
14) }
```



6) Squeeze: removing a character from a string in place. This C program compiles and runs. What is its output?

```
1) #include <stdio.h>
2)
 3) void main(void)
 4) {
 5)
      char s[]="XbXXytXXXe";
 6)
      char del ='X';
      int srcIdx=0, snkIdx=0;
7)
8)
      while (s[srcIdx])
      { if (s[srcIdx] != del)
9)
        { s[snkIdx] = s[srcIdx];
10)
11)
          snkIdx++;
        }
12)
13)
        else
14)
        { printf("[%d,%d] %s\n", srcIdx, snkIdx, s);
15)
16)
        srcIdx++;
17)
18)
      s[snkIdx]='\0';
      printf("==>%s\n",s);
19)
20) }
```



7) This C program compiles and runs. What is its output?

```
1)
      #include <stdio.h>
 2)
 3)
     void main(void)
 4)
     {
 5)
 6)
        char bits[40];
        bits[39] = '\0';
 7)
        unsigned int n=400;
 8)
        unsigned int p;
 9)
10)
        int i;
11)
        int k=38;
        for (i=0; i<32; i++)</pre>
12)
13)
          unsigned int p=1<<i;</pre>
14)
          if (n & p) bits[k] = '1';
15)
          else bits[k] = '0';
16)
          if ((i+1) % 4 == 0) bits[--k] = '-';
17)
18)
          k--;
        }
19)
20)
21)
       printf("%s\n", bits);
22)
```