Keep this test booklet when you are done. **Closed book. No notes.** Work strictly from memory. **No calculators. Scratch paper okay.**

Section I: Record all answers on the bubble sheet.

On the following printf questions you are given a list of inputs. For each problem line determine which printf statement created the accompanying outputs. (␣ means space.)

Which of these printf statements created the outputs shown for each problem below? (`x` is `int x;`)

(A) `printf("␣␣␣␣␣␣%␣3d␣",x);` (D) `printf("␣␣␣%02d␣␣␣␣␣",x);` (G) `printf("%+10d",x);`
(B) `printf("␣␣␣␣%0+6d",x);` (E) `printf("␣␣%+d␣␣␣␣␣␣␣␣",x);` (H) `printf("%-8d␣␣",x);`
(C) `printf("␣␣␣␣%d␣␣␣␣␣",x);` (F) `printf("␣␣%-2d␣␣␣␣␣␣",x);` (I) `printf("%4d␣␣␣␣␣␣",x);`

inputs:	2	-6	1218095657	-1788544269
1/2p.	␣␣␣2␣␣␣␣␣␣	␣␣-6␣␣␣␣␣␣	1218095657␣␣␣␣␣␣	-1788544269␣␣␣␣␣␣
2/2p.	2␣␣␣␣␣␣␣␣␣	-6␣␣␣␣␣␣␣␣	1218095657␣␣	-1788544269␣␣
3/2p.	␣␣␣␣+00002	␣␣␣␣-00006	␣␣␣␣+1218095657	␣␣␣␣-1788544269
4/2p.	␣␣␣␣2␣␣␣␣␣	␣␣␣␣-6␣␣␣␣␣	␣␣␣␣1218095657␣␣␣␣␣	␣␣␣␣-1788544269␣␣␣␣␣

Which of these printf statements created the outputs shown for each problem below? (`x` is `char * x;`)

(A) `printf("␣␣␣␣%s␣␣",x);` (D) `printf("␣%2s␣␣␣",x);` (G) `printf("␣%s␣␣␣␣␣",x);`
(B) `printf("␣␣%1s␣␣␣",x);` (E) `printf("␣%4s␣",x);` (H) `printf("%-3s␣␣␣",x);`
(C) `printf("␣%-4s␣",x);` (F) `printf("␣%5s",x);` (I) `printf("%-6s",x);`

inputs:	""	"f"	"lv"	"qfpq"	"wplqld"	"wzzhwppj"
5/2p.	␣␣␣␣␣␣	f␣␣␣␣␣	lv␣␣␣␣	qfpq␣␣␣	wplqld␣␣␣	wzzhwppj␣␣␣
6/2p.	␣␣␣␣␣␣	␣␣␣␣␣f	␣␣␣␣lv	␣␣qfpq	␣wplqld	␣wzzhwppj
7/2p.	␣␣␣␣␣␣	␣␣f␣␣␣	␣␣lv␣␣␣	␣␣qfpq␣␣␣	␣␣wplqld␣␣␣	␣␣wzzhwppj␣␣␣
8/2p.	␣␣␣␣␣␣	␣f␣␣␣␣	␣lv␣␣␣	␣qfpq␣	␣wplqld␣	␣wzzhwppj␣

Which of these printf statements created the outputs shown for each problem below? (`x` is `double x;`)

(A) `printf("␣␣␣%+011.4f",x);` (D) `printf("␣␣%11.4f␣",x);` (G) `printf("␣%12.4f␣",x);`
(B) `printf("␣␣%0+12f",x);` (E) `printf("␣%+10.2f␣␣␣",x);` (H) `printf("␣%13.0f",x);`
(C) `printf("␣␣%011.6f␣",x);` (F) `printf("␣%0+10.4f␣␣␣",x);` (I) `printf("%+14.6f",x);`

inputs:	0	-1.81	-4.2180	-147110.289769
9/2p.	␣␣␣␣␣␣␣0.0000␣	␣␣␣␣␣␣␣-1.8100␣	␣␣␣␣␣␣␣-4.2180␣	␣-147110.2898␣
10/2p.	␣␣␣␣␣␣␣0.0000␣	␣␣␣␣␣␣␣-1.8100␣	␣␣␣␣␣␣␣-4.2180␣	␣␣-147110.2898␣
11/2p.	␣␣␣+00000.0000	␣␣␣-00001.8100	␣␣␣-00004.2180	␣␣␣-147110.2898
12/2p.	␣␣␣␣␣␣␣␣␣␣␣␣␣0	␣␣␣␣␣␣␣␣␣␣␣␣␣-2	␣␣␣␣␣␣␣␣␣␣␣␣␣-4	␣␣␣␣␣␣␣-147110

Precedence: What is the value of each expression? Mark (I) for error, (J) for none of the above.

13/1p.	`6*3<=1!=3*4`	(A) 0	(B) 1	(C) 4	(D) 6	(E) 24	(F) 26	(G) 40	(H) 93
14/1p.	`9-9/9-5/9`	(A) -64	(B) -9	(C) -1	(D) 1	(E) 8	(F) 9	(G) 10	(H) 98
15/1p.	`4*3+0&&0<4`	(A) -89	(B) -87	(C) -41	(D) 0	(E) 1	(F) 4	(G) 12	(H) 29
16/1p.	`2%9%9-5-2`	(A) -20	(B) -5	(C) -2	(D) 0	(E) 2	(F) 30	(G) 38	(H) 40
17/1p.	`9+4>=8==0*9`	(A) -77	(B) -73	(C) 0	(D) 1	(E) 9	(F) 10	(G) 77	(H) 90
18/1p.	`6/2%5-6+6`	(A) -23	(B) -9	(C) -4	(D) -2	(E) 1	(F) 3	(G) 4	(H) 6
19/1p.	`1/4-6+6*7`	(A) -84	(B) -48	(C) -7	(D) -1	(E) 0	(F) 35	(G) 36	(H) 48
20/1p.	`7-5*5*5%2`	(A) -94	(B) -28	(C) -18	(D) 0	(E) 2	(F) 10	(G) 47	(H) 98
21/1p.	`2-9*3/3*2`	(A) -18	(B) -16	(C) -14	(D) -5	(E) -4	(F) -2	(G) 0	(H) 19
22/1p.	`7*7==0<=1+1`	(A) -94	(B) -12	(C) 0	(D) 1	(E) 2	(F) 8	(G) 14	(H) 91
23/1p.	`1+5-7/3*7`	(A) -63	(B) -7	(C) -6	(D) -1	(E) 0	(F) 6	(G) 28	(H) 72
24/1p.	`9-6-1-9-9`	(A) -84	(B) -38	(C) -16	(D) -14	(E) 2	(F) 4	(G) 20	(H) 22
25/1p.	`2*6-7>=9!=0`	(A) -85	(B) -63	(C) -25	(D) 1	(E) 2	(F) 10	(G) 12	(H) 76
26/1p.	`6/3/1/3-9`	(A) -87	(B) -77	(C) -72	(D) -9	(E) -3	(F) -2	(G) 28	(H) 66
27/1p.	`2+9%7-9-8`	(A) -75	(B) -51	(C) -13	(D) 1	(E) 3	(F) 5	(G) 36	(H) 49
28/1p.	`1/9+6-7+5`	(A) -12	(B) -6	(C) -2	(D) 0	(E) 5	(F) 11	(G) 81	(H) 84
29/1p.	`4-6%2*6+6`	(A) -8	(B) 4	(C) 6	(D) 10	(E) 16	(F) 22	(G) 30	(H) 48
30/1p.	`0>=0&&6+8*2`	(A) -81	(B) -77	(C) -74	(D) 0	(E) 1	(F) 2	(G) 16	(H) 18
31/1p.	`4+9*0/8-4`	(A) -52	(B) -32	(C) -4	(D) 0	(E) 1	(F) 4	(G) 28	(H) 80

How many times does the body of the loop execute? (Mark 9 if 9 or more.)

32/2p. `int i; for( i=-1; i<-1; i++ ) body;`

33/2p. `int z=0; while( ++z != -5 ) body;`

34/2p. `int b; for( b=9; b<11; b++ ) body;`

35/2p. `int a; for( a=-9; a>=-15; --a ) body;`

36/2p. `int z; for( z=5; z<=11; z-- ) body;`

37/2p. `int b=-9; while( --b <= -3 ) body;`

38/2p. `int u; for( u=3; u>-4; --u ) body;`

39/2p. `int n; for( n=4; n<7; ++n ) body;`

40/3p. Give a tight big-oh $\Theta()$ bound on the running time T(n) of this program.
Assume `atoi`, `simpleStatement`, and `simpleCompare` each run in $\Theta(1)$ time.
(A) $n^2\sqrt{n}$ (C) $n\sqrt{n}$ (E) n (G) $\sqrt{n}$ (I) $\lg n$
(B) $n^2 \lg n$ (D) $n \lg n$ (F) $\sqrt{n}\lg n$ (H) $\lg^2 n$ (J) 1

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   if ( simpleCompare ) {
      b = 1; do {
         simpleStatement;
      b++; } while ( b * b < n );
   } else {
      simpleStatement;
   }
   return 0; }
```

41/3p. Give a tight big-oh $\Theta()$ bound on the running time T(n) of this program.
Assume `atoi`, `simpleStatement`, and `simpleCompare` each run in $\Theta(1)$ time.
(A) $n^2\sqrt{n}$ (C) n^2 (E) $n\sqrt{n}$ (G) n (I) $\lg n$
(B) $n^2 \lg n$ (D) $n\sqrt{n}\lg n$ (F) $n \lg n$ (H) $\lg^2 n$ (J) 1

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   if ( simpleCompare ) {
      for ( g = 1 ; g * g < n ; g += 2 ) {
         h = 1; while ( h * h < n ) {
            simpleStatement;
         h += 3; }
      }
   } else {
      simpleStatement;
   }
   return 0; }
```

42/3p. Give a tight big-oh $\Theta()$ bound on the running time T(n) of this program.
Assume `atoi`, `simpleStatement`, and `simpleCompare` each run in $\Theta(1)$ time.
(A) $n^2\sqrt{n}$ (C) $n\sqrt{n}\lg n$ (E) $n \lg n$ (G) $\sqrt{n}\lg n$ (I) $\lg n$
(B) $n^2 \lg n$ (D) $n\sqrt{n}$ (F) n (H) $\lg^2 n$ (J) 1

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   if ( simpleCompare ) {
      if ( simpleCompare ) {
         simpleStatement;
      } else {
         simpleStatement;
      }
   } else {
      simpleStatement;
   }
   return 0; }
```

43/5p. Give a tight big-oh $\Theta()$ bound on the running time T(n) of this program.
Assume `atoi`, `simpleStatement`, and `simpleCompare` each run in $\Theta(1)$ time.
(A) n^2 (C) $n\sqrt{n}$ (E) n (G) $\sqrt{n}$ (I) $\lg n$
(B) $n\sqrt{n}\lg n$ (D) $n\lg n$ (F) $\sqrt{n}\lg n$ (H) $\lg^2 n$ (J) 1

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   for ( h = 1 ; h * h < n ; h += 1 ) {
      d = 1; do {
         if ( simpleCompare ) {
            if ( simpleCompare ) {
               simpleStatement;
            } else {
               simpleStatement;
            }
         }
      d *= 2; } while ( d < n );
   }
   return 0; }
```

44/5p. Give a tight big-oh $\Theta()$ bound on the running time T(n) of this program.
Assume `atoi`, `simpleStatement`, and `simpleCompare` each run in $\Theta(1)$ time.
(A) $n^2\sqrt{n}$ (C) n^2 (E) $n\lg n$ (G) $\sqrt{n}\lg n$ (I) $\lg n$
(B) $n^2\lg n$ (D) $n\sqrt{n}\lg n$ (F) n (H) $\lg^2 n$ (J) 1

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   if ( simpleCompare ) {
      if ( simpleCompare ) {
         for ( a = 1 ; a < n ; a *= 2 ) {
            if ( simpleCompare ) {
               simpleStatement;
            } else {
               simpleStatement;
            }
         }
      } else {
         if ( simpleCompare ) {
            simpleStatement;
         }
      }
   } else {
      g = 1; while ( g * g < n ) {
         i = 1; do {
            simpleStatement;
         i++; } while ( i * i < n );
      g++; }
   }
   return 0; }
```

45/10p. Give a tight big-oh $\Theta()$ bound on the running time T(n) of this program.
Assume `atoi`, `simpleStatement`, and `simpleCompare` each run in $\Theta(1)$ time.

(A) n^3 (C) n^2 (E) $n\sqrt{n}$ (G) n (I) $\lg^2 n$
(B) $n^2 \lg n$ (D) $n\sqrt{n}\lg n$ (F) $n \lg n$ (H) $\sqrt{n}$ (J) $\lg n$

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   if ( simpleCompare ) {
      if ( simpleCompare ) {
         for ( c = 1 ; c < n ; c++ ) {
            h = 1; while ( h < n ) {
               if ( simpleCompare ) {
                  j = 1; do {
                     simpleStatement;
                  j++; } while ( j * j < n );
               } else {
                  simpleStatement;
               }
            h *= 2; }
         }
      } else {
         if ( simpleCompare ) {
            if ( simpleCompare ) {
               simpleStatement;
            } else {
               simpleStatement;
            }
         } else {
            simpleStatement;
         }
      }
   } else {
      k = 1; while ( k * k < n ) {
         a = 1; do {
            b = n; do {
               if ( simpleCompare ) {
                  if ( simpleCompare ) {
                     simpleStatement;
                  } else {
                     simpleStatement;
                  }
               }
            b /= 2; } while ( b > 1 );
         a += 10; } while ( a * a < n );
      k += 2; }
   }
   return 0; }
```

Section II: List regex answers on a single well-labeled sheet of paper.

46/4p. Let L be the language accepted by the regular expression "`(b|aa)?a?`". List the shortest five (or all) strings in L.

47/4p. Let L be the language accepted by the regular expression "`(aab|ab)+ba`". List the shortest five (or all) strings in L.

48/4p. Let L be the language accepted by the regular expression "`(a?|ba)bab?`". List the shortest five (or all) strings in L.

49/4p. Let L be the language accepted by the regular expression "`(bba(bab|baa)+baa(abb|aab)*aab(aba|b*)?)+`". List the shortest five (or all) strings in L.

Section III: Write each program on a separate well-labeled sheet of paper.

50/8p. Write this program on a separate sheet of paper. Use one side of one sheet. Label it clearly. Accept two integers, a and b, from the command line (argv). Print the larger (max).

51/8p. Write this program on a separate sheet of paper. Use one side of one sheet. Label it clearly. Prompt for and accept an integer. Tell whether it is a multiple of 2 or not.

52/15p. Write this program on a separate sheet of paper. Use one side of one sheet. Label it clearly. Add all numbers on the command line (argv). Print the answer.

53/15p. Write this program on a separate sheet of paper. Use one side of one sheet. Label it clearly. Prompt for and accept a start time and an ending time in HH:MM:SS format. Print the difference. Example: start 1:39:44, end 3:51:15, difference 2:11:31.

Total points 150.

Answer Key (points per line)

1	(2).	I
2	(2).	H
3	(2).	B
4	(2).	C
5	(2).	H
6	(2).	F
7	(2).	B
8	(2).	C
9	(2).	G
10	(2).	D
11	(2).	A
12	(2).	H
13	(1).	B (1)
14	(1).	E (8)
15	(1).	E (1)
16	(1).	B (-5)
17	(1).	C (0)
18	(1).	F (3)
19	(1).	G (36)
20	(1).	J (6)
21	(1).	B (-16)
22	(1).	C (0)
23	(1).	J (-8)
24	(1).	C (-16)
25	(1).	J (0)
26	(1).	D (-9)
27	(1).	C (-13)
28	(1).	J (4)
29	(1).	D (10)
30	(1).	E (1)
31	(1).	D (0)
32	(2).	0
33	(2).	9
34	(2).	2
35	(2).	7
36	(2).	9
37	(2).	9
38	(2).	7
39	(2).	3
40	(3).	G ($\sqrt{n}$)
41	(3).	G (n)
42	(3).	J (1)
43	(5).	F ($\sqrt{n}\lg n$)
44	(5).	F (n)
45	(10).	D ($n\sqrt{n}\lg n$)

Total points 150.