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("␣%+5d␣",x);` (D) `printf("%+07d",x);` (G) `printf("%-+7d",x);`
(B) `printf("␣%+d␣␣␣␣␣",x);` (E) `printf("%+7d",x);` (H) `printf("%0␣6d␣",x);`
(C) `printf("␣%-3d␣␣␣",x);` (F) `printf("%-+6d␣",x);` (I) `printf("%0␣7d",x);`

inputs:	8	-3	1339825118	-2010112755
1/2p.	`␣␣␣␣␣+8`	`␣␣␣␣␣-3`	`+1339825118`	`-2010112755`
2/2p.	`+8␣␣␣␣␣`	`-3␣␣␣␣␣`	`+1339825118␣`	`-2010112755␣`
3/2p.	`␣00008␣`	`-00003␣`	`␣1339825118␣`	`-2010112755␣`
4/2p.	`␣000008`	`-000003`	`␣1339825118`	`-2010112755`

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

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

inputs:	""	"p"	"zq"	"gdfw"	"zzdccg"	"fwpcpkwj"
5/2p.	`␣␣␣␣␣`	`p␣␣␣␣`	`zq␣␣␣`	`gdfw␣`	`zzdccg␣`	`fwpcpkwj␣`
6/2p.	`␣␣␣␣␣`	`␣␣␣p␣`	`␣␣zq␣`	`␣gdfw␣`	`␣zzdccg␣`	`␣fwpcpkwj␣`
7/2p.	`␣␣␣␣␣`	`␣␣␣␣p`	`␣␣␣zq`	`␣gdfw`	`zzdccg`	`fwpcpkwj`
8/2p.	`␣␣␣␣␣`	`␣␣p␣␣`	`␣␣zq␣␣`	`␣␣gdfw␣␣`	`␣␣zzdccg␣␣`	`␣␣fwpcpkwj␣␣`

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

(A) `printf("␣␣%+11.0f␣",x);` (D) `printf("␣%+13f",x);` (G) `printf("␣%11.6f␣␣",x);`
(B) `printf("␣␣%+12f",x);` (E) `printf("␣%010f␣␣␣",x);` (H) `printf("%+14.6f",x);`
(C) `printf("␣␣%12f",x);` (F) `printf("␣%013.2f",x);` (I) `printf("%13.6f␣",x);`

inputs:	7	1.68	4.5305	-646588.565890
9/2p.	`␣␣␣␣␣+7.000000`	`␣␣␣␣␣+1.680000`	`␣␣␣␣␣+4.530500`	`-646588.565890`
10/2p.	`␣␣␣␣␣␣␣␣␣␣␣+7␣`	`␣␣␣␣␣␣␣␣␣␣␣+2␣`	`␣␣␣␣␣␣␣␣␣␣␣+5␣`	`␣␣␣␣␣␣-646589␣`
11/2p.	`␣␣␣␣␣7.000000␣`	`␣␣␣␣␣1.680000␣`	`␣␣␣␣␣4.530500␣`	`-646588.565890␣`
12/2p.	`␣007.000000␣␣␣`	`␣001.680000␣␣␣`	`␣004.530500␣␣␣`	`␣-646588.565890␣␣␣`

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

13/1p.	`8-6%3/4+7`	(A) -75	(B) -59	(C) -49	(D) 0	(E) 1	(F) 8	(G) 9	(H) 15
14/1p.	`3*9/4/3-4`	(A) -43	(B) -9	(C) -6	(D) -4	(E) -2	(F) 15	(G) 23	(H) 41
15/1p.	`3*1!=2&&3*7`	(A) -6	(B) -5	(C) 0	(D) 1	(E) 7	(F) 21	(G) 59	(H) 60
16/1p.	`4%8+1/5+2`	(A) 0	(B) 1	(C) 2	(D) 3	(E) 4	(F) 6	(G) 12	(H) 46
17/1p.	`3/8<6<=9*7`	(A) -59	(B) -45	(C) 0	(D) 1	(E) 3	(F) 7	(G) 21	(H) 78
18/1p.	`3+3-1/1*8`	(A) -69	(B) -2	(C) 0	(D) 6	(E) 19	(F) 40	(G) 43	(H) 73
19/1p.	`4-6*0+4-0`	(A) -61	(B) -20	(C) 0	(D) 4	(E) 15	(F) 41	(G) 43	(H) 83
20/1p.	`6*9%2-6-8`	(A) -78	(B) -14	(C) -10	(D) -6	(E) 2	(F) 6	(G) 8	(H) 18
21/1p.	`6+3%5/5-9`	(A) -8	(B) -7	(C) -3	(D) 1	(E) 5	(F) 15	(G) 19	(H) 21
22/1p.	`6*4%8/9+5`	(A) -41	(B) -16	(C) 1	(D) 4	(E) 5	(F) 24	(G) 30	(H) 73
23/1p.	`9*7>=2||7-4`	(A) -3	(B) 1	(C) 5	(D) 9	(E) 32	(F) 43	(G) 65	(H) 99
24/1p.	`0*2+8-1+8`	(A) -83	(B) -40	(C) -10	(D) -8	(E) -1	(F) 0	(G) 7	(H) 8
25/1p.	`7-6*7/1*2`	(A) -77	(B) -70	(C) -18	(D) -14	(E) -11	(F) 3	(G) 76	(H) 78
26/1p.	`1-0+7%2+9`	(A) -65	(B) -26	(C) -9	(D) -6	(E) 5	(F) 11	(G) 28	(H) 52
27/1p.	`6+0<=6<=9-4`	(A) -41	(B) -19	(C) -4	(D) -3	(E) 0	(F) 1	(G) 3	(H) 64
28/1p.	`2-7%9-2*2`	(A) -14	(B) -9	(C) -8	(D) -6	(E) -5	(F) -1	(G) 2	(H) 9
29/1p.	`1-3+8>=3>=6`	(A) -49	(B) -24	(C) -3	(D) -1	(E) 1	(F) 46	(G) 51	(H) 62
30/1p.	`5*9>8==4+5`	(A) -32	(B) 0	(C) 5	(D) 6	(E) 10	(F) 25	(G) 36	(H) 65
31/1p.	`2/8%9-5-1`	(A) -4	(B) -2	(C) -1	(D) 0	(E) 1	(F) 13	(G) 21	(H) 90

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

32/2p. `int b=7; do body; while( --b >= 0 );`

33/2p. `int w; for( w=10; w>6; --w ) body;`

34/2p. `int h; for( h=-8; h>=-12; --h ) body;`

35/2p. `int q=1; do body; while( q-- > -6 );`

36/2p. `int k=0; while( --k > -5 ) body;`

37/2p. `int p=2; while( ++p != -2 ) body;`

38/2p. `int y=-8; do body; while( y-- != -8 );`

39/2p. `int g=-7; while( ++g != -3 ) 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^2 (E) $n\sqrt{n}$ (G) $\sqrt{n}$ (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 ) {
      for ( b = 1 ; b * b < n ; b += 2 ) {
         simpleStatement;
      }
   } 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) $\sqrt{n}\lg n$ (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 ) {
      j = 1; do {
         c = n; do {
            simpleStatement;
         c /= 2; } while ( c > 1 );
      j += 1; } while ( j < n );
   } 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^2 (E) $n\lg^2 n$ (G) $\sqrt{n}\lg n$ (I) $\lg^3 n$
(B) $n^2 \lg n$ (D) $n\sqrt{n}$ (F) $n \lg n$ (H) $\sqrt{n}$ (J) 1

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   i = 1; do {
      for ( g = 1 ; g < n ; g++ ) {
         h = 1; while ( h < n ) {
            if ( simpleCompare ) {
               simpleStatement;
            }
         h *= 2; }
      }
   i *= 2; } while ( i < n );
   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^3 (C) n^2 (E) $n\sqrt{n}$ (G) n (I) $\lg^3 n$
(B) $n^2\sqrt{n}$ (D) $n\sqrt{n}\lg n$ (F) $n\lg n$ (H) $\sqrt{n}$ (J) $\lg^2 n$

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   for ( d = 1 ; d * d < n ; d += 10 ) {
      for ( e = 1 ; e < n ; e++ ) {
         f = 1; while ( f < n ) {
            if ( simpleCompare ) {
               if ( simpleCompare ) {
                  simpleStatement;
               }
            } else {
               simpleStatement;
            }
         f += 10; }
      }
   }
   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^4 (C) $n^3\lg n$ (E) $n^2\sqrt{n}$ (G) n^2 (I) $\sqrt{n}\lg^4 n$
(B) $n^3\sqrt{n}$ (D) n^3 (F) $n^2\lg^3 n$ (H) $n\sqrt{n}\lg^2 n$ (J) 1

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   f = 1; while ( f * f < n ) {
      e = n; do {
         i = 1; do {
            if ( simpleCompare ) {
               j = 1; do {
                  b = n; do {
                     simpleStatement;
                  b /= 3; } while ( b > 1 );
               j += 2; } while ( j * j < n );
            } else {
               if ( simpleCompare ) {
                  simpleStatement;
               } else {
                  simpleStatement;
               }
            }
         i++; } while ( i * i < n );
      e /= 2; } while ( e > 1 );
   f += 10; }
   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) $n^2\sqrt{n}$ (C) $n\sqrt{n}\lg n$ (E) $n\lg^2 n$ (G) $\sqrt{n}\lg n$ (I) $\lg^2 n$
(B) n^2 (D) $n\sqrt{n}$ (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 ) {
         if ( simpleCompare ) {
            if ( simpleCompare ) {
               if ( simpleCompare ) {
                  simpleStatement;
               } else {
                  simpleStatement;
               }
            } else {
               simpleStatement;
            }
         } else {
            simpleStatement;
         }
      } else {
         if ( simpleCompare ) {
            b = 1; while ( b < n ) {
               simpleStatement;
            b += 2; }
         } else {
            simpleStatement;
         }
      }
   } else {
      for ( d = n ; d > 1 ; d-- ) {
         for ( c = n ; c > 1 ; c /= 2 ) {
            f = 1; while ( f * f < n ) {
               if ( simpleCompare ) {
                  if ( simpleCompare ) {
                     simpleStatement;
                  }
               } else {
                  simpleStatement;
               }
            f += 1; }
         }
      }
   }
   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 "`(bababa)aa?`". List the shortest five (or all) strings in L.

47/4p. Let L be the language accepted by the regular expression "`abb(bab|b*)?`". List the shortest five (or all) strings in L.

48/4p. Let L be the language accepted by the regular expression "`(bbaaba)+`". List the shortest five (or all) strings in L.

49/4p. Let L be the language accepted by the regular expression "`(abab)a*??a*bbabab*bab(aaa)?`". 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 their quotient (a divided by b).

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. Prompt for and accept a string. Print out its letters in reverse order.

53/15p. Write this program on a separate sheet of paper. Use one side of one sheet. Label it clearly. Write a subroutine that, given a word, returns 'yes' if its vowels are in alphabetical order, otherwise returns 'no'. Vowels are a, e, i, o, and u.

Total points 150.

Answer Key (points per line)

1	(2).	E
2	(2).	F
3	(2).	H
4	(2).	I
5	(2).	G
6	(2).	F
7	(2).	I
8	(2).	B
9	(2).	H
10	(2).	A
11	(2).	I
12	(2).	E
13	(1).	H (15)
14	(1).	E (-2)
15	(1).	D (1)
16	(1).	F (6)
17	(1).	D (1)
18	(1).	B (-2)
19	(1).	J (8)
20	(1).	B (-14)
21	(1).	C (-3)
22	(1).	E (5)
23	(1).	B (1)
24	(1).	J (15)
25	(1).	A (-77)
26	(1).	F (11)
27	(1).	F (1)
28	(1).	B (-9)
29	(1).	J (0)
30	(1).	B (0)
31	(1).	J (-6)
32	(2).	8
33	(2).	4
34	(2).	5
35	(2).	8
36	(2).	4
37	(2).	9
38	(2).	1
39	(2).	3
40	(3).	G ($\sqrt{n}$)
41	(3).	F ($n \lg n$)
42	(3).	E ($n \lg^2 n$)
43	(5).	B ($n^2\sqrt{n}$)
44	(5).	H ($n\sqrt{n} \lg^2 n$)
45	(10).	C ($n\sqrt{n} \lg n$)

Total points 150.