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("␣␣␣␣%+4d␣␣",x);`
(B) `printf("␣␣␣␣%+d␣␣␣␣␣",x);`
(C) `printf("␣␣␣␣%-␣3d␣␣␣",x);`
(D) `printf("␣␣␣␣%0␣4d␣␣",x);`
(E) `printf("␣␣␣%+-5d␣␣",x);`
(F) `printf("␣␣␣%6d␣",x);`
(G) `printf("␣␣%+5d␣␣␣",x);`
(H) `printf("␣%05d␣␣␣␣",x);`
(I) `printf("%-10d",x);`

inputs:	2	-6	1766149591	-1222285792
1/2p.	␣␣␣␣+2␣␣␣␣␣	␣␣␣␣-6␣␣␣␣␣	␣␣␣␣+1766149591␣␣␣␣␣	␣␣␣␣-1222285792␣␣␣␣␣
2/2p.	␣␣␣␣␣␣␣␣2␣	␣␣␣␣␣␣␣-6␣	␣␣␣1766149591␣	␣␣␣-1222285792␣
3/2p.	␣␣␣␣␣+2␣␣␣	␣␣␣␣␣-6␣␣␣	␣␣+1766149591␣␣␣	␣␣-1222285792␣␣␣
4/2p.	␣␣␣␣␣2␣␣␣␣	␣␣␣␣-6␣␣␣␣	␣␣␣␣␣1766149591␣␣␣	␣␣␣␣-1222285792␣␣␣

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

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

inputs:	“”	“p”	“cz”	“fpyb”	“xdlzcl”	“xxvwfgff”
5/2p.	␣␣␣␣␣	␣␣p␣␣	␣␣cz␣␣	␣␣fpyb␣␣	␣␣xdlzcl␣␣	␣␣xxvwfgff␣␣
6/2p.	␣␣␣␣␣	␣␣␣p␣	␣␣␣cz	␣␣␣fpyb	␣␣␣xdlzcl	␣␣␣xxvwfgff
7/2p.	␣␣␣␣␣	␣␣␣p␣␣	␣␣␣cz␣␣	␣␣␣fpyb␣␣	␣␣␣xdlzcl␣␣	␣␣␣xxvwfgff␣␣
8/2p.	␣␣␣␣␣	p␣␣␣␣	cz␣␣␣	fpyb␣	xdlzcl	xxvwfgff

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

(A) `printf("␣␣␣%0+10f␣",x);`
(B) `printf("␣␣␣%11.6f",x);`
(C) `printf("␣␣%+12.2f",x);`
(D) `printf("␣␣%010.2f␣␣",x);`
(E) `printf("%+012.0f␣␣",x);`
(F) `printf("%+014.4f",x);`
(G) `printf("%+14f",x);`
(H) `printf("%12.4f␣␣",x);`
(I) `printf("%14.6f",x);`

inputs:	5	1.50	-3.7828	-92787.463538
9/2p.	␣␣␣␣␣␣5.000000	␣␣␣␣␣␣1.500000	␣␣␣␣␣-3.782800	␣␣␣-92787.463538
10/2p.	␣␣0000005.00␣␣	␣␣0000001.50␣␣	␣␣-000003.78␣␣	␣␣-092787.46␣␣
11/2p.	␣␣␣␣␣␣␣␣+5.00	␣␣␣␣␣␣␣␣+1.50	␣␣␣␣␣␣␣␣-3.78	␣␣␣␣␣-92787.46
12/2p.	␣␣␣␣␣␣5.000000	␣␣␣␣␣␣1.500000	␣␣␣␣␣-3.782800	␣-92787.463538

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

13/1p.	`1-8%1+9*6`	(A) -53	(B) -47	(C) -7	(D) 48	(E) 53	(F) 54	(G) 55	(H) 60
14/1p.	`4/1!=6&&6*7`	(A) -88	(B) -27	(C) 1	(D) 4	(E) 7	(F) 20	(G) 76	(H) 91
15/1p.	`4+5-6%6*5`	(A) -60	(B) 3	(C) 15	(D) 29	(E) 45	(F) 76	(G) 79	(H) 94
16/1p.	`2%2+2/4-8`	(A) -97	(B) -95	(C) -39	(D) -5	(E) -4	(F) 0	(G) 4	(H) 94
17/1p.	`2-9/7+8%3`	(A) -88	(B) -22	(C) -1	(D) 0	(E) 2	(F) 3	(G) 8	(H) 26
18/1p.	`4-5%9/7%3`	(A) -41	(B) -21	(C) 0	(D) 1	(E) 2	(F) 4	(G) 8	(H) 37
19/1p.	`5*6/9+1*4`	(A) -78	(B) -59	(C) 0	(D) 4	(E) 7	(F) 12	(G) 16	(H) 20
20/1p.	`5-0*3+8/6`	(A) -1	(B) 0	(C) 2	(D) 4	(E) 5	(F) 6	(G) 16	(H) 20
21/1p.	`4-9==1>2+1`	(A) -12	(B) 0	(C) 1	(D) 2	(E) 3	(F) 5	(G) 7	(H) 50
22/1p.	`9+6<2<9/2`	(A) 0	(B) 4	(C) 5	(D) 9	(E) 10	(F) 30	(G) 56	(H) 79
23/1p.	`4-6/3*4+8`	(A) -20	(B) -4	(C) -1	(D) 4	(E) 12	(F) 16	(G) 24	(H) 77
24/1p.	`4*7>9&&9-5`	(A) -78	(B) -16	(C) -5	(D) -4	(E) -1	(F) 0	(G) 1	(H) 45
25/1p.	`0-7-9/4/7`	(A) -7	(B) -2	(C) 0	(D) 1	(E) 15	(F) 67	(G) 81	(H) 95
26/1p.	`9/7-8%3%6`	(A) -60	(B) -53	(C) -49	(D) -1	(E) 0	(F) 1	(G) 4	(H) 5
27/1p.	`3*8%5+7%6`	(A) -95	(B) -67	(C) -53	(D) 0	(E) 4	(F) 5	(G) 10	(H) 12
28/1p.	`4+1>3==4-8`	(A) -82	(B) -8	(C) -7	(D) -4	(E) -3	(F) 0	(G) 1	(H) 5
29/1p.	`6%6*6-8+5`	(A) -36	(B) -7	(C) -3	(D) 0	(E) 3	(F) 5	(G) 6	(H) 11
30/1p.	`4%8&&3&&2+8`	(A) -46	(B) -18	(C) 0	(D) 1	(E) 4	(F) 9	(G) 54	(H) 97
31/1p.	`6+7-6%4%5`	(A) -98	(B) -63	(C) -15	(D) 2	(E) 3	(F) 6	(G) 7	(H) 11

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

32/2p. `int z=5; while( z++ <= 11 ) body;`

33/2p. `int x=-4; while( --x >= -8 ) body;`

34/2p. `int s=-9; while( --s >= -11 ) body;`

35/2p. `int u=1; while( u++ <= 3 ) body;`

36/2p. `int m; for( m=-4; m>-5; --m ) body;`

37/2p. `int q; for( q=-2; q<5; ++q ) body;`

38/2p. `int i; for( i=8; i!=12; i++ ) body;`

39/2p. `int d; for( d=-1; d!=-7; d-- ) 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) n (I) $\lg n$
(B) $n^2 \lg n$ (D) $n\sqrt{n}\lg n$ (F) $n \lg n$ (H) $\sqrt{n}$ (J) 1

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   if ( simpleCompare ) {
      for ( f = n ; f > 1 ; f /= 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) $n\sqrt{n}\lg n$ (E) $n \lg n$ (G) $\sqrt{n}\lg n$ (I) $\lg^2 n$
(B) n^2 (D) $n\sqrt{n}$ (F) n (H) $\sqrt{n}$ (J) 1

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

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

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

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

45/5p. 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) $n^5\sqrt{n}\lg n$ (C) $n^3\sqrt{n}$ (E) $n^2\lg^4 n$ (G) $n\sqrt{n}\lg^5 n$ (I) $\sqrt{n}\lg^6 n$
(B) $n^4\sqrt{n}$ (D) $n^3\lg^2 n$ (F) n^2 (H) $n\sqrt{n}\lg^4 n$ (J) $\lg^2 n$

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   k = n; do {
      i = 1; do {
         f = n; while ( f > 1 ) {
            b = n; while ( b > 1 ) {
               if ( simpleCompare ) {
                  if ( simpleCompare ) {
                     if ( simpleCompare ) {
                        if ( simpleCompare ) {
                           if ( simpleCompare ) {
                              simpleStatement;
                           } else {
                              simpleStatement;
                           }
                        } else {
                           simpleStatement;
                        }
                     } else {
                        simpleStatement;
                     }
                  } else {
                     e = 1; while ( e * e < n ) {
                        h = 1; do {
                           simpleStatement;
                        h *= 5; } while ( h < n );
                     e++; }
                  }
               } else {
                  for ( c = n ; c > 1 ; c /= 5 ) {
                     a = 1; while ( a * a < n ) {
                        if ( simpleCompare ) {
                           for ( j = 1 ; j * j < n ; j++ ) {
                              simpleStatement;
                           }
                        } else {
                           simpleStatement;
                        }
                     a += 10; }
                  }
               }
            b--; }
         f /= 2; }
      i *= 3; } while ( i < n );
   k /= 3; } while ( k > 1 );
   return 0; }
```

Matching: Which Perl regular expression commands have what meaning? (If no match mark J.)

(A) `$` (B) `*` (C) `+` (D) `\S` (E) `\W` (F) `\s` (G) `\w` (H) `^` (I) `{`

46/1p. whitespace
47/1p. repeat zero or more times
48/1p. non a word character
49/1p. start of character class
50/1p. end of string

True or False: does the string match the regular expression?

51/1p. Does the string "`bdnu`" match the regular expression "`(bd)?nu`"?

52/1p. Does the empty string match the regular expression "`c+cb`"?

53/1p. Does the string "`pppp`" match the regular expression "`pp+|kg+`"?

54/1p. Does the string "`fc`" match the regular expression "`(ta)*fc`"?

55/1p. Does the string "`uzz`" match the regular expression "`uz+q*`"?

56/1p. Does the string "`y`" match the regular expression "`yy?|nt+|c`"?

57/1p. Does the string "`tbw`" match the regular expression "`(xr)?tb?w`"?

58/1p. Does the string "`tt`" match the regular expression "`dg+qd|u?|tt`"?

59/1p. Does the string "`sfnf`" match the regular expression "`kw*(sk)*|(fn?)*`"?

60/1p. Does the string "`rgx`" match the regular expression "`rg|xp*`"?

61/1p. Does the string "`ppabns`" match the regular expression "`p*((au)*)?|bn?|s`"?

62/1p. Does the string "`ysktyskt`" match the regular expression "`p*zh|((ys)+kt)*`"?

Section II: Non-Bubble Answers: Write neatly. Use the paper provided. Answer on the front ONLY of each page. On the upper-right corner of the back, lightly write your 7-digit BYUH student id number. Do NOT write your NAME anywhere. (This aids fairness in grading.) Leave margins the same size as on this page. Sloppy/non-compliant work will be penalized.

Write each program on a separate sheet of paper. Write the program number in the upper left corner and circle it.

63/10p. Write this program: Prompt for and accept two integers, a and b. Print their product (multiply).

64/10p. Write this program: Accept a list of one or more integers from the command line (argv). Print those evenly divisible by 3, one per line.

65/10p. Write this program: Prompt for and accept a number n. Print its factors (positive numbers that divide into it evenly), smallest to largest. For 12, print 1 2 3 4 6 12.

66/10p. Write this program: Adding time. Prompt for and accept a start time and a running time in HH:MM:SS format. Print the ending time. Example: 1:39:44 + 2:11:31 = 3:51:15.

67/10p. Write this program: Write a subroutine that, given a list of words, returns an alphabetical list of the distinct words with a count of each. Sample input: a b a x c c a a d b, output: a=4 b=2 c=2 d=1 x=1.

Total points 150.

Answer Key (points per line)

1 (2).	B	32 (2).	7
2 (2).	F	33 (2).	4
3 (2).	G	34 (2).	2
4 (2).	C	35 (2).	3
5 (2).	C	36 (2).	1
6 (2).	A	37 (2).	7
7 (2).	B	38 (2).	4
8 (2).	H	39 (2).	6
9 (2).	B	40 (3).	I ($\lg n$)
10 (2).	D	41 (3).	D ($n\sqrt{n}$)
11 (2).	C	42 (3).	F ($\sqrt{n}\lg n$)
12 (2).	I	43 (5).	B ($n^3\sqrt{n}$)
13 (1).	G (55)	44 (5).	C ($n\sqrt{n}\lg n$)
14 (1).	C (1)	45 (5).	E ($n^2\lg^4 n$)
15 (1).	J (9)	46 (1).	F
16 (1).	J (-8)	47 (1).	B
17 (1).	F (3)	48 (1).	E
18 (1).	F (4)	49 (1).	J
19 (1).	E (7)	50 (1).	A
20 (1).	F (6)	51 (1).	true
21 (1).	B (0)	52 (1).	false
22 (1).	J (1)	53 (1).	true
23 (1).	D (4)	54 (1).	true
24 (1).	G (1)	55 (1).	true
25 (1).	A (-7)	56 (1).	true
26 (1).	D (-1)	57 (1).	true
27 (1).	F (5)	58 (1).	true
28 (1).	F (0)	59 (1).	false
29 (1).	C (-3)	60 (1).	false
30 (1).	D (1)	61 (1).	false
31 (1).	H (11)	62 (1).	true

Total points 150.