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);`
(B) `printf("␣␣␣␣%02d␣␣␣␣",x);`
(C) `printf("␣%+04d␣␣␣␣␣",x);`
(D) `printf("␣%-8d␣",x);`
(E) `printf("%+09d␣",x);`
(F) `printf("%+10d",x);`
(G) `printf("%0␣10d",x);`
(H) `printf("%0+10d",x);`
(I) `printf("%10d",x);`

inputs:	1	-7	1639031896	-1514616187
1/2p.	␣␣␣␣01␣␣␣␣	␣␣␣␣-7␣␣␣␣	␣␣␣␣1639031896␣␣␣␣	␣␣␣␣-1514616187␣␣␣␣
2/2p.	␣␣␣␣␣␣␣␣+1	␣␣␣␣␣␣␣␣-7	+1639031896	-1514616187
3/2p.	+00000001␣	-00000007␣	+1639031896␣	-1514616187␣
4/2p.	␣␣␣␣1␣␣␣␣␣	␣␣␣␣-7␣␣␣␣	␣␣␣␣1639031896␣␣␣	␣␣␣␣-1514616187␣␣␣

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

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

inputs:	""	"y"	"qd"	"ybvl"	"kqfvgl"	"gbczypxj"
5/2p.	␣␣␣␣␣␣␣	␣y␣␣␣␣␣	␣qd␣␣␣␣␣	␣ybvl␣␣␣␣␣	␣kqfvgl␣␣␣␣␣	␣gbczypxj␣␣␣␣␣
6/2p.	␣␣␣␣␣␣␣	␣␣␣␣␣y␣	␣␣␣␣qd␣	␣␣ybvl␣	␣kqfvgl␣	␣gbczypxj␣
7/2p.	␣␣␣␣␣␣␣	␣␣␣␣y␣␣	␣␣␣␣qd␣␣	␣␣␣␣ybvl␣␣	␣␣␣␣kqfvgl␣␣	␣␣␣␣gbczypxj␣␣
8/2p.	␣␣␣␣␣␣␣	␣y␣␣␣␣␣	␣qd␣␣␣␣	␣ybvl␣␣	␣kqfvgl␣␣	␣gbczypxj␣␣

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

(A) `printf("␣␣␣%10.0f␣",x);`
(B) `printf("␣␣%+011f␣",x);`
(C) `printf("␣␣%012.0f",x);`
(D) `printf("␣␣%11f␣",x);`
(E) `printf("␣%+013.0f",x);`
(F) `printf("␣%+13.2f",x);`
(G) `printf("␣%0+12.6f␣",x);`
(H) `printf("␣%10.6f␣␣␣",x);`
(I) `printf("%11.6f␣␣␣",x);`

inputs:	9	-3.26	2.5799	-34326.691431
9/2p.	␣␣000000000009	␣␣-00000000003	␣␣000000000003	␣␣-00000034327
10/2p.	␣+000000000009	␣-000000000003	␣+000000000003	␣-000000034327
11/2p.	␣␣␣9.000000␣␣␣	␣␣-3.260000␣␣␣	␣␣␣2.579900␣␣␣	-34326.691431␣␣␣
12/2p.	␣+0009.000000␣	␣-0003.260000␣	␣+0002.579900␣	␣-34326.691431␣

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

13/1p.	`3*7%7-4+3`	(A) -9	(B) -7	(C) -3	(D) -1	(E) 3	(F) 12	(G) 13	(H) 44
14/1p.	`3-7-2-7*4`	(A) -52	(B) -34	(C) -30	(D) -24	(E) 4	(F) 11	(G) 16	(H) 20
15/1p.	`7-7%1+1*7`	(A) -55	(B) -1	(C) 7	(D) 14	(E) 56	(F) 81	(G) 94	(H) 95
16/1p.	`4*9>=5&&6+5`	(A) -68	(B) -22	(C) 1	(D) 6	(E) 9	(F) 24	(G) 58	(H) 74
17/1p.	`8*7-1%6/4`	(A) -21	(B) 0	(C) 8	(D) 12	(E) 13	(F) 22	(G) 30	(H) 56
18/1p.	`5*6%9*2%5`	(A) -33	(B) 0	(C) 1	(D) 5	(E) 6	(F) 10	(G) 12	(H) 30
19/1p.	`4/0||9||8+7`	(A) -91	(B) -71	(C) -59	(D) -58	(E) 4	(F) 8	(G) 11	(H) 47
20/1p.	`6/4+9%4+7`	(A) 0	(B) 3	(C) 8	(D) 9	(E) 10	(F) 13	(G) 51	(H) 78
21/1p.	`1+9>=4==2-3`	(A) -89	(B) -38	(C) -32	(D) -3	(E) -2	(F) -1	(G) 1	(H) 2
22/1p.	`5/2-9/2-4`	(A) -7	(B) -6	(C) -5	(D) 0	(E) 1	(F) 2	(G) 3	(H) 7
23/1p.	`6<5!=1-7%8`	(A) -6	(B) -1	(C) 0	(D) 1	(E) 2	(F) 16	(G) 46	(H) 72
24/1p.	`4/0+9/9-3`	(A) -3	(B) -2	(C) 0	(D) 4	(E) 23	(F) 59	(G) 60	(H) 79
25/1p.	`2<7<7+4*4`	(A) -97	(B) -80	(C) 0	(D) 1	(E) 4	(F) 16	(G) 20	(H) 42
26/1p.	`8/2%3-0+3`	(A) -84	(B) -39	(C) -2	(D) 1	(E) 4	(F) 7	(G) 49	(H) 76
27/1p.	`8+5-4/8+4`	(A) -85	(B) 5	(C) 8	(D) 12	(E) 13	(F) 17	(G) 81	(H) 88
28/1p.	`7*1/9+9-3`	(A) -74	(B) -66	(C) -21	(D) -3	(E) 0	(F) 6	(G) 40	(H) 42
29/1p.	`7+0+8*2/8`	(A) -37	(B) -35	(C) -33	(D) 0	(E) 3	(F) 7	(G) 8	(H) 59
30/1p.	`7/3&&9<=6*8`	(A) -91	(B) -19	(C) 1	(D) 7	(E) 8	(F) 40	(G) 56	(H) 67
31/1p.	`4*2%4*9+3`	(A) 3	(B) 8	(C) 9	(D) 11	(E) 20	(F) 62	(G) 75	(H) 96

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

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

33/2p. `int s; for( s=-9; s!=-3; s-- ) body;`

34/2p. `int b=-8; while( b-- > -9 ) body;`

35/2p. `int u=-8; do body; while( u++ <= -3 );`

36/2p. `int a; for( a=8; a>=4; a++ ) body;`

37/2p. `int z=6; do body; while( ++z != 11 );`

38/2p. `int x=-1; while( x++ <= 5 ) body;`

39/2p. `int f=9; do body; while( --f != 4 );`

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 (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]);
   if ( simpleCompare ) {
      k = 1; while ( k * k < n ) {
         simpleStatement;
      k++; }
   } 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) $n\lg n$ (G) $\sqrt{n}\lg n$ (I) $\lg^2 n$
(B) $n^2\lg n$ (D) $n\sqrt{n}$ (F) n (H) $\sqrt{n}$ (J) 1

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   if ( simpleCompare ) {
      d = 1; do {
         j = 1; do {
            simpleStatement;
         j++; } while ( j < n );
      d++; } while ( d * d < 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) $n^2\lg n$ (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) $\lg n$

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   h = n; do {
      if ( simpleCompare ) {
         g = n; do {
            simpleStatement;
         g /= 2; } while ( g > 1 );
      } else {
         simpleStatement;
      }
   h -= 10; } while ( h > 1 );
   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) $n^2\sqrt{n}$ (C) $n\sqrt{n}$ (E) n (G) $\sqrt{n}$ (I) $\lg n$
(B) n^2 (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 ( a = 1 ; a * a < n ; a++ ) {
      if ( simpleCompare ) {
         if ( simpleCompare ) {
            simpleStatement;
         } else {
            simpleStatement;
         }
      } else {
         simpleStatement;
      }
   }
   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^3 (C) $n^2 \lg n$ (E) $n\sqrt{n}$ (G) n (I) $\sqrt{n} \lg n$
(B) $n^2\sqrt{n}$ (D) n^2 (F) $n \lg^2 n$ (H) $\sqrt{n} \lg^2 n$ (J) $\sqrt{n}$

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

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

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

(A) `\B` (B) `\W` (C) `\b` (D) `\i` (E) `\n` (F) `\nn` (G) `\t` (H) `\w` (I) `\xnn`

46/1p. hex character nn
47/1p. carriage return
48/1p. non a word character
49/1p. tab
50/1p. word boundary

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

51/1p. Does the empty string match the regular expression "`x*a`"?

52/1p. Does the string "`qfcw`" match the regular expression "`(qf)+|cw`"?

53/1p. Does the string "`ww`" match the regular expression "`(cx)*|w+`"?

54/1p. Does the string "`ps`" match the regular expression "`ps*|(kc)+`"?

55/1p. Does the string "`bbr`" match the regular expression "`b?|r|qz`"?

56/1p. Does the string "`wkqwwtp`" match the regular expression "`wk?qw+tp`"?

57/1p. Does the empty string match the regular expression "`(w*|bb)?|(kt)*`"?

58/1p. Does the string "`zr`" match the regular expression "`(wq)?|zr(ht)?`"?

59/1p. Does the string "`ncnchpayay`" match the regular expression "`(nc)*|(hp)*p(ay)+`"?

60/1p. Does the string "`yuuquchhun`" match the regular expression "`(yu)+|qu|ch?un`"?

61/1p. Does the string "`zbqzr`" match the regular expression "`gw|zb(q+zr+)?`"?

62/1p. Does the string "`qcqhdz`" match the regular expression "`q+cq(hd)*z`"?

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 remainder (a divided by b).

64/10p. Write this program: Accept a list of one or more integers from the command line (argv). Print the largest (max).

65/10p. Write this program: Given n on the command line, print all integers from 2 to n-1 that divide evenly into n (factors). For 12 print 2 3 4 6

66/10p. Write this program: 5

67/10p. Write this program: 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).	B	32 (2).	1
2 (2).	F	33 (2).	9
3 (2).	E	34 (2).	1
4 (2).	A	35 (2).	7
5 (2).	G	36 (2).	9
6 (2).	H	37 (2).	5
7 (2).	A	38 (2).	7
8 (2).	E	39 (2).	5
9 (2).	C	40 (3).	G ($\sqrt{n}$)
10 (2).	E	41 (3).	D ($n\sqrt{n}$)
11 (2).	I	42 (3).	E ($n \lg n$)
12 (2).	G	43 (5).	G ($\sqrt{n}$)
13 (1).	D (-1)	44 (5).	H ($\sqrt{n} \lg^2 n$)
14 (1).	B (-34)	45 (5).	H ($\sqrt{n} \lg n$)
15 (1).	D (14)	46 (1).	I
16 (1).	C (1)	47 (1).	J
17 (1).	H (56)	48 (1).	B
18 (1).	C (1)	49 (1).	G
19 (1).	I (error)	50 (1).	C
20 (1).	D (9)	51 (1).	false
21 (1).	J (0)	52 (1).	false
22 (1).	B (-6)	53 (1).	true
23 (1).	D (1)	54 (1).	true
24 (1).	I (error)	55 (1).	false
25 (1).	D (1)	56 (1).	true
26 (1).	E (4)	57 (1).	true
27 (1).	F (17)	58 (1).	true
28 (1).	F (6)	59 (1).	false
29 (1).	J (9)	60 (1).	false
30 (1).	C (1)	61 (1).	true
31 (1).	A (3)	62 (1).	true

Total points 150.