Do NOT write on this test. Record all answers on the bubble sheet. **Closed book. No notes.** Work strictly from memory. **No calculators. No time limit. Scratch paper okay.**

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

1/1p.	`8%1-3+8-6`	(A) -17	(B) -9	(C) -8	(D) -5	(E) -1	(F) 0	(G) 2	(H) 33
2/1p.	`1+9%4%6-5`	(A) -94	(B) -70	(C) -51	(D) -3	(E) 0	(F) 1	(G) 10	(H) 66
3/1p.	`8-0/2+0/9`	(A) -81	(B) -59	(C) -44	(D) -20	(E) -10	(F) 0	(G) 4	(H) 64
4/1p.	`4-6/9-0*5`	(A) -63	(B) -56	(C) -11	(D) -5	(E) -1	(F) 4	(G) 20	(H) 83
5/1p.	`6*2/8-3/9`	(A) -98	(B) -84	(C) -12	(D) -6	(E) -2	(F) 1	(G) 6	(H) 61
6/1p.	`7-9+6*6-9`	(A) -92	(B) -74	(C) -57	(D) -47	(E) -20	(F) -12	(G) 25	(H) 52
7/1p.	`5*1-8%9%9`	(A) -87	(B) -35	(C) -3	(D) 1	(E) 6	(F) 10	(G) 23	(H) 61
8/1p.	`4*3-6-0%5`	(A) -12	(B) -8	(C) 1	(D) 3	(E) 6	(F) 8	(G) 11	(H) 34
9/1p.	`7%4/5-6%4`	(A) -35	(B) -2	(C) -1	(D) 0	(E) 1	(F) 2	(G) 3	(H) 75
10/1p.	`5/5/3+4*1`	(A) -84	(B) -56	(C) -55	(D) 0	(E) 1	(F) 4	(G) 35	(H) 86
11/1p.	`9/0-8%7-7`	(A) -32	(B) -16	(C) -9	(D) -1	(E) 1	(F) 2	(G) 87	(H) 95
12/1p.	`2*5&&4<4-6`	(A) -12	(B) -10	(C) -6	(D) -5	(E) -4	(F) 0	(G) 1	(H) 86
13/1p.	`3%1-6+8/6`	(A) -7	(B) -5	(C) -3	(D) -2	(E) -1	(F) 1	(G) 46	(H) 65
14/1p.	`7+7/9%9+1`	(A) -55	(B) 2	(C) 7	(D) 8	(E) 14	(F) 47	(G) 61	(H) 85
15/1p.	`5*9<8==3-3`	(A) -89	(B) -59	(C) -46	(D) -15	(E) -3	(F) 1	(G) 5	(H) 98
16/1p.	`2-8+1<1<=8`	(A) -7	(B) -5	(C) 1	(D) 2	(E) 19	(F) 31	(G) 82	(H) 99
17/1p.	`2>=5&&0-0-3`	(A) -86	(B) -84	(C) -3	(D) -2	(E) 0	(F) 3	(G) 86	(H) 88
18/1p.	`5+2%7%6-6`	(A) -67	(B) -25	(C) -6	(D) -5	(E) -1	(F) 1	(G) 2	(H) 90
19/1p.	`4-5-9+1%4`	(A) -79	(B) -36	(C) -11	(D) -3	(E) 1	(F) 7	(G) 9	(H) 79
20/1p.	`8+1<=5||5*6`	(A) -86	(B) -45	(C) 1	(D) 6	(E) 9	(F) 10	(G) 14	(H) 54
21/1p.	`6*1==5!=6+4`	(A) 0	(B) 1	(C) 4	(D) 5	(E) 6	(F) 10	(G) 56	(H) 89
22/1p.	`4-0-7/5/4`	(A) -74	(B) -3	(C) -1	(D) 0	(E) 1	(F) 4	(G) 11	(H) 66
23/1p.	`3*7+0==0>=4`	(A) -60	(B) -41	(C) 0	(D) 1	(E) 3	(F) 21	(G) 22	(H) 60
24/1p.	`7/6!=0!=3+9`	(A) -72	(B) -48	(C) -22	(D) 0	(E) 9	(F) 10	(G) 16	(H) 50
25/1p.	`3*8-0/8+9`	(A) -42	(B) -3	(C) 0	(D) 1	(E) 12	(F) 15	(G) 30	(H) 51
26/1p.	`8+4/2/2%5`	(A) -48	(B) 0	(C) 2	(D) 3	(E) 5	(F) 9	(G) 12	(H) 88
27/1p.	`4/9/2-2*9`	(A) -17	(B) -11	(C) -9	(D) -4	(E) -1	(F) 0	(G) 18	(H) 27
28/1p.	`1%1<=3>=3-8`	(A) -51	(B) -17	(C) -6	(D) -4	(E) 0	(F) 1	(G) 49	(H) 74
29/1p.	`0<=5<=4-3*4`	(A) -11	(B) -8	(C) 0	(D) 1	(E) 4	(F) 48	(G) 74	(H) 75
30/1p.	`9/6!=7||2-1`	(A) -1	(B) 0	(C) 8	(D) 38	(E) 53	(F) 56	(G) 72	(H) 87
31/1p.	`7+1/3-9-4`	(A) -97	(B) -83	(C) -11	(D) -6	(E) -4	(F) -3	(G) -1	(H) 2
32/1p.	`8-2-4<=2!=4`	(A) -99	(B) -58	(C) 0	(D) 1	(E) 5	(F) 7	(G) 66	(H) 96
33/1p.	`8/5%2%5*2`	(A) -42	(B) 1	(C) 2	(D) 4	(E) 6	(F) 8	(G) 16	(H) 32
34/1p.	`7-9-0>9<=8`	(A) -69	(B) -3	(C) -2	(D) -1	(E) 1	(F) 6	(G) 7	(H) 47

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

(A) `$` (B) `(` (C) `\2` (D) `\W` (E) `\n` (F) `\r` (G) `\s` (H) `\w` (I) `^`

35/1p. carriage return
36/1p. start of group
37/1p. end of string
38/1p. second back-reference
39/1p. word character

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

(A) `$` (B) `)` (C) `\B` (D) `\D` (E) `\b` (F) `\nn` (G) `\xnn` (H) `eog` (I) `|`

40/1p. word boundary
41/1p. digit
42/1p. hex character nn
43/1p. not a word boundary
44/1p. end of group

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

45/1p. Does the string "`pwpwqg`" match the regular expression "`(pw)?qg`"?

46/1p. Does the string "`pggf`" match the regular expression "`pg*|f`"?

47/1p. Does the string "`ffgy`" match the regular expression "`f?gy`"?

48/1p. Does the string "`bn`" match the regular expression "`bn+|xg`"?

49/1p. Does the empty string match the regular expression "`(n|zx)+`"?

50/1p. Does the string "`dq`" match the regular expression "`(bq)*|dq|dg`"?

51/1p. Does the string "`aa`" match the regular expression "`a+a?`"?

52/1p. Does the string "`cxkff`" match the regular expression "`cx(kf)+`"?

53/1p. Does the string "`ah`" match the regular expression "`wp*|(ah+)+`"?

54/1p. Does the string "`fu`" match the regular expression "`d*fu|xy`"?

55/1p. Does the string "`t`" match the regular expression "`gd+|(sb)+|t`"?

56/1p. Does the string "`gsgssqzz`" match the regular expression "`(gs)*sq(zz)?`"?

57/1p. Does the string "`pxcqr`" match the regular expression "`px?|c|qr`"?

58/1p. Does the string "`t`" match the regular expression "`f?t*|(uk)+`"?

59/1p. Does the string "`ww`" match the regular expression "`w+|z*|(xg)*`"?

60/1p. Does the string "`nyqz`" match the regular expression "`nh*|y|qf+z`"?

61/1p. Does the string "`dsbwzshsh`" match the regular expression "`ds+b*wz(sh)+`"?

62/1p. Does the empty string match the regular expression "`(((aq)+|(sp)*)+a)?`"?

63/1p. Does the string "`yryr`" match the regular expression "`(yr)+gn+|(zu)+`"?

64/1p. Does the string "`pyay`" match the regular expression "`a+|py?ay`"?

65/1p. Does the string "`xpxpyycchqhq`" match the regular expression "`(xp)*(y+|c)?hq?`"?

66/1p. Does the string "`ysysghqq`" match the regular expression "`ys+|gh((qr)*)+`"?

67/1p. Does the string "`gzgzdn`" match the regular expression "`gz?|d+|yb|(n*)*`"?

68/1p. Does the string "`s`" match the regular expression "`c?s(xg?dh|r)*`"?

69/2p. 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) $n^2\sqrt{n}\lg^2 n$ (C) $n^2\sqrt{n}$ (E) $n^2\lg^2 n$ (G) $n\lg^3 n$ (I) $\lg^2 n$
(B) $n^2\sqrt{n}\lg n$ (D) $n^2\lg^3 n$ (F) $n^2\lg n$ (H) $\sqrt{n}\lg^4 n$ (J) 1

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

70/2p. 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 ) {
      for ( c = 1 ; c < n ; c *= 2 ) {
         if ( simpleCompare ) {
            if ( simpleCompare ) {
               simpleStatement;
            } else {
               simpleStatement;
            }
         }
      }
   } else {
      if ( simpleCompare ) {
         for ( k = 1 ; k * k < n ; k += 3 ) {
            simpleStatement;
         }
      }
   }
   return 0; }
```

71/2p. 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]);
   b = 1; while ( b * b < n ) {
      if ( simpleCompare ) {
         if ( simpleCompare ) {
            simpleStatement;
         } else {
            simpleStatement;
         }
      } else {
         simpleStatement;
      }
   b++; }
   return 0; }
```

72/2p. 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) $n^2\sqrt{n}$ (E) $n\sqrt{n}\lg^2 n$ (G) $n\sqrt{n}$ (I) $\sqrt{n}\lg^3 n$
(B) $n^2\sqrt{n}\lg n$ (D) n^2 (F) $n\sqrt{n}\lg n$ (H) $n\lg^2 n$ (J) $\lg^2 n$

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   if ( simpleCompare ) {
      for ( g = 1 ; g < n ; g *= 5 ) {
         for ( k = 1 ; k < n ; k *= 5 ) {
            if ( simpleCompare ) {
               c = n; while ( c > 1 ) {
                  h = 1; while ( h * h < n ) {
                     if ( simpleCompare ) {
                        simpleStatement;
                     } else {
                        simpleStatement;
                     }
                  h++; }
               c--; }
            } else {
               i = 1; while ( i * i < n ) {
                  simpleStatement;
               i += 5; }
            }
         }
      }
   }
   return 0; }
```

73/2p. 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 (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 ) {
      for ( k = 1 ; k < n ; k += 10 ) {
         if ( simpleCompare ) {
            simpleStatement;
         } else {
            simpleStatement;
         }
      }
   } else {
      simpleStatement;
   }
   return 0; }
```

74/2p. 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) $n^3\sqrt{n}$ (C) $n\sqrt{n} \lg n$ (E) $n \lg^3 n$ (G) n (I) $\sqrt{n}$
(B) $n^2\sqrt{n}$ (D) $n\sqrt{n}$ (F) $n \lg^2 n$ (H) $\sqrt{n} \lg n$ (J) $\lg^2 n$

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

75/2p. 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 (G) $\sqrt{n}$ (I) $\lg n$
(B) $n^2 \lg n$ (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]);
   if ( simpleCompare ) {
      c = 1; while ( c < n ) {
         if ( simpleCompare ) {
            if ( simpleCompare ) {
               simpleStatement;
            } else {
               simpleStatement;
            }
         } else {
            simpleStatement;
         }
      c++; }
   }
   return 0; }
```

76/2p. 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) $n^3\sqrt{n}$ (C) $n^2\sqrt{n}$ (E) n^2 (G) $n\sqrt{n}$ (I) $\sqrt{n}\lg^3 n$
(B) $n^2\sqrt{n}\lg n$ (D) $n^2 \lg n$ (F) $n\sqrt{n}\lg n$ (H) n (J) 1

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

77/2p. 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) $n \lg n$ (I) $\lg^2 n$
(B) $n^2 \lg n$ (D) $n\sqrt{n} \lg n$ (F) $n \lg^2 n$ (H) $\sqrt{n}$ (J) 1

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   i = n; while ( i > 1 ) {
      if ( simpleCompare ) {
         for ( f = n ; f > 1 ; f /= 5 ) {
            if ( simpleCompare ) {
               for ( e = 1 ; e < n ; e += 5 ) {
                  simpleStatement;
               }
            }
         }
      } else {
         d = 1; while ( d < n ) {
            simpleStatement;
         d *= 5; }
      }
   i /= 5; }
   return 0; }
```

78/2p. 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 (G) $\sqrt{n}$ (I) $\lg n$
(B) $n^2 \lg n$ (D) $n\sqrt{n} \lg n$ (F) $\sqrt{n} \lg n$ (H) $\lg^2 n$ (J) 1

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

79/2p. 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) $\sqrt{n}$ (I) $\lg n$
(B) $n^2\lg n$ (D) $n\lg^2 n$ (F) n (H) $\lg^2 n$ (J) 1

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   a = 1; do {
      if ( simpleCompare ) {
         g = n; while ( g > 1 ) {
            if ( simpleCompare ) {
               d = 1; do {
                  simpleStatement;
               d++; } while ( d * d < n );
            }
         g -= 5; }
      } else {
         i = 1; while ( i * i < n ) {
            simpleStatement;
         i += 3; }
      }
   a += 2; } while ( a < n );
   return 0; }
```

80/2p. 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) $\sqrt{n}$ (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 ) {
         if ( simpleCompare ) {
            for ( d = 1 ; d < n ; d *= 3 ) {
               simpleStatement;
            }
         }
      } else {
         simpleStatement;
      }
   } else {
      if ( simpleCompare ) {
         simpleStatement;
      } else {
         simpleStatement;
      }
   }
   return 0; }
```

81/2p. 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) $n^3\sqrt{n}$ (C) $n^2\sqrt{n}$ (E) n^2 (G) $n\sqrt{n}$ (I) n
(B) $n^2\sqrt{n}\lg n$ (D) $n^2\lg n$ (F) $n\sqrt{n}\lg n$ (H) $n\lg^2 n$ (J) $\lg^4 n$

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

82/2p. 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) $n \lg^2 n$ (G) $\sqrt{n} \lg n$ (I) $\lg^2 n$
(B) n^2 (D) $n\sqrt{n}$ (F) $n \lg n$ (H) $\sqrt{n}$ (J) 1

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

83/2p. 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) $\sqrt{n}$ (I) $\lg^2 n$
(B) $n^2 \lg n$ (D) $n\sqrt{n} \lg n$ (F) $\sqrt{n} \lg^2 n$ (H) $\lg^3 n$ (J) 1

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

84/2p. 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) $n^2 \lg n$ (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]);
   if ( simpleCompare ) {
      if ( simpleCompare ) {
         if ( simpleCompare ) {
            if ( simpleCompare ) {
               if ( simpleCompare ) {
                  if ( simpleCompare ) {
                     if ( simpleCompare ) {
                        if ( simpleCompare ) {
                           simpleStatement;
                        } else {
                           simpleStatement;
                        }
                     }
                  }
               } else {
                  if ( simpleCompare ) {
                     if ( simpleCompare ) {
                        simpleStatement;
                     } else {
                        simpleStatement;
                     }
                  }
               }
            } else {
               if ( simpleCompare ) {
                  if ( simpleCompare ) {
                     if ( simpleCompare ) {
                        for ( g = n ; g > 1 ; g /= 2 ) {
                           simpleStatement;
                        }
                     } else {
                        simpleStatement;
                     }
                  } else {
                     c = n; do {
                        simpleStatement;
                     c--; } while ( c > 1 );
                  }
               }
            }
         }
   }
   return 0; }
```

Total points 100.

Answer Key (points per line)

1 (1).	E (-1)	43 (1).	C
2 (1).	D (-3)	44 (1).	B
3 (1).	J (8)	45 (1).	false
4 (1).	F (4)	46 (1).	false
5 (1).	F (1)	47 (1).	false
6 (1).	G (25)	48 (1).	true
7 (1).	C (-3)	49 (1).	false
8 (1).	E (6)	50 (1).	true
9 (1).	B (-2)	51 (1).	true
10 (1).	F (4)	52 (1).	false
11 (1).	I (error)	53 (1).	true
12 (1).	F (0)	54 (1).	true
13 (1).	B (-5)	55 (1).	true
14 (1).	D (8)	56 (1).	true
15 (1).	F (1)	57 (1).	false
16 (1).	C (1)	58 (1).	true
17 (1).	E (0)	59 (1).	true
18 (1).	F (1)	60 (1).	false
19 (1).	J (-9)	61 (1).	true
20 (1).	C (1)	62 (1).	true
21 (1).	B (1)	63 (1).	false
22 (1).	F (4)	64 (1).	true
23 (1).	C (0)	65 (1).	false
24 (1).	J (1)	66 (1).	false
25 (1).	J (33)	67 (1).	false
26 (1).	F (9)	68 (1).	true
27 (1).	J (-18)	69 (2).	G ($n \lg^3 n$)
28 (1).	F (1)	70 (2).	G ($\sqrt{n}$)
29 (1).	C (0)	71 (2).	G ($\sqrt{n}$)
30 (1).	J (1)	72 (2).	E ($n\sqrt{n} \lg^2 n$)
31 (1).	D (-6)	73 (2).	E (n)
32 (1).	D (1)	74 (2).	A ($n^3\sqrt{n}$)
33 (1).	C (2)	75 (2).	E (n)
34 (1).	E (1)	76 (2).	B ($n^2\sqrt{n} \lg n$)
35 (1).	F	77 (2).	F ($n \lg^2 n$)
36 (1).	B	78 (2).	I ($\lg n$)
37 (1).	A	79 (2).	A ($n^2\sqrt{n}$)
38 (1).	C	80 (2).	I ($\lg n$)
39 (1).	H	81 (2).	D ($n^2 \lg n$)
40 (1).	E	82 (2).	B (n^2)
41 (1).	J	83 (2).	D ($n\sqrt{n} \lg n$)
42 (1).	G	84 (2).	E (n)

Total points 100.