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.	`9-4-5+2+8`	(A) -10	(B) 4	(C) 6	(D) 10	(E) 16	(F) 20	(G) 75	(H) 77
2/1p.	`0+7+3%5*9`	(A) -41	(B) -38	(C) 0	(D) 10	(E) 25	(F) 34	(G) 85	(H) 90
3/1p.	`8+5+3-6*2`	(A) -43	(B) 4	(C) 7	(D) 9	(E) 12	(F) 20	(G) 55	(H) 58
4/1p.	`2%6%8%4-7`	(A) -7	(B) -5	(C) -3	(D) 0	(E) 1	(F) 10	(G) 35	(H) 96
5/1p.	`7/9-0/4-7`	(A) -74	(B) -7	(C) -4	(D) -2	(E) 0	(F) 3	(G) 7	(H) 70
6/1p.	`5-8/6-0-8`	(A) -42	(B) -4	(C) -1	(D) 5	(E) 7	(F) 9	(G) 12	(H) 30
7/1p.	`1/6-7*4/5`	(A) -51	(B) -5	(C) -1	(D) 0	(E) 2	(F) 12	(G) 17	(H) 48
8/1p.	`3-9%1%3+1`	(A) -97	(B) 0	(C) 1	(D) 2	(E) 4	(F) 48	(G) 59	(H) 66
9/1p.	`7%6-0*9+4`	(A) -58	(B) -3	(C) 1	(D) 5	(E) 7	(F) 11	(G) 13	(H) 25
10/1p.	`2/3-7%2*3`	(A) -86	(B) -40	(C) -35	(D) -3	(E) -1	(F) 1	(G) 3	(H) 5
11/1p.	`0-8%2-7*5`	(A) -35	(B) -19	(C) -15	(D) 10	(E) 17	(F) 35	(G) 69	(H) 93
12/1p.	`5-8+9/9*8`	(A) -67	(B) -16	(C) -11	(D) -3	(E) -1	(F) 0	(G) 32	(H) 40
13/1p.	`6*8%4+9%6`	(A) -89	(B) -56	(C) 3	(D) 6	(E) 12	(F) 18	(G) 90	(H) 95
14/1p.	`3+5%6*0-3`	(A) -97	(B) -16	(C) -12	(D) -10	(E) -6	(F) -1	(G) 0	(H) 2
15/1p.	`8-4==9>=2-1`	(A) -76	(B) -1	(C) 0	(D) 1	(E) 7	(F) 8	(G) 9	(H) 60
16/1p.	`9%8%2%7-9`	(A) -10	(B) -8	(C) -3	(D) -1	(E) 73	(F) 77	(G) 80	(H) 81
17/1p.	`9/7%4/2+7`	(A) -99	(B) -8	(C) 0	(D) 7	(E) 8	(F) 10	(G) 16	(H) 79
18/1p.	`1%2-6/6%4`	(A) -17	(B) -3	(C) -2	(D) -1	(E) 1	(F) 22	(G) 75	(H) 90
19/1p.	`9*4>0||6+4`	(A) -91	(B) -26	(C) 0	(D) 1	(E) 5	(F) 13	(G) 29	(H) 54
20/1p.	`7-5*3+4*2`	(A) -56	(B) -48	(C) -31	(D) -8	(E) 0	(F) 14	(G) 20	(H) 22
21/1p.	`8*7*1/2*3`	(A) -66	(B) -65	(C) -49	(D) 0	(E) 8	(F) 9	(G) 72	(H) 87
22/1p.	`5+9+5>5==8`	(A) -86	(B) -49	(C) -28	(D) 1	(E) 5	(F) 6	(G) 14	(H) 19
23/1p.	`6/5%4/2*8`	(A) -47	(B) 1	(C) 8	(D) 24	(E) 33	(F) 37	(G) 48	(H) 73
24/1p.	`3>3<4-6-4`	(A) -71	(B) -67	(C) -9	(D) -4	(E) -3	(F) -1	(G) 0	(H) 1
25/1p.	`7+3>3&&4*8`	(A) -88	(B) -24	(C) 1	(D) 7	(E) 8	(F) 56	(G) 64	(H) 65
26/1p.	`4+7-7>=0==0`	(A) -59	(B) -43	(C) 0	(D) 1	(E) 4	(F) 11	(G) 30	(H) 71
27/1p.	`4+5-0==7!=3`	(A) -74	(B) -27	(C) 0	(D) 4	(E) 5	(F) 8	(G) 64	(H) 69
28/1p.	`0-8-0%7%6`	(A) -91	(B) -5	(C) -2	(D) 0	(E) 4	(F) 5	(G) 33	(H) 86
29/1p.	`3+3!=5>4*8`	(A) 0	(B) 1	(C) 4	(D) 5	(E) 8	(F) 11	(G) 24	(H) 32
30/1p.	`6/3>=8<9*4`	(A) -17	(B) 0	(C) 1	(D) 4	(E) 6	(F) 24	(G) 54	(H) 64
31/1p.	`8*6&&3&&7+3`	(A) 1	(B) 4	(C) 8	(D) 11	(E) 32	(F) 37	(G) 66	(H) 79
32/1p.	`4+7<2&&3*6`	(A) -87	(B) -28	(C) -4	(D) 0	(E) 4	(F) 6	(G) 11	(H) 24
33/1p.	`8<0<5-0+4`	(A) -49	(B) -3	(C) 0	(D) 1	(E) 4	(F) 5	(G) 36	(H) 73
34/1p.	`3+3/7+3*9`	(A) -83	(B) -77	(C) 0	(D) 3	(E) 10	(F) 27	(G) 30	(H) 54

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

(A) `)` (B) `*` (C) `+` (D) `\B` (E) `\D` (F) `\b` (G) `\d` (H) `\nn` (I) `\xnn`

35/1p. repeat zero or more times
36/1p. word boundary
37/1p. non-digit
38/1p. repeat one or more times
39/1p. hex character nn

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

(A) `$` (B) `C-X` (C) `\` (D) `\D` (E) `\S` (F) `\cX` (G) `\s` (H) `^` (I) `{,n,}`

40/1p. start of string
41/1p. exactly n times
42/1p. escape the next character
43/1p. control X
44/1p. whitespace

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

45/1p. Does the string "`q`" match the regular expression "`xr?q`"?

46/1p. Does the string "`cq`" match the regular expression "`(cq)*|dt`"?

47/1p. Does the string "`u`" match the regular expression "`(uw)*|qx`"?

48/1p. Does the string "`dzb`" match the regular expression "`(dz)+b`"?

49/1p. Does the string "`nnaa`" match the regular expression "`n*(na)*`"?

50/1p. Does the string "`w`" match the regular expression "`w*|kp|ks`"?

51/1p. Does the empty string match the regular expression "`c?|dp`"?

52/1p. Does the string "`ca`" match the regular expression "`ca|(gu)*`"?

53/1p. Does the string "`rr`" match the regular expression "`r?|h(zh)+`"?

54/1p. Does the string "`rrpa`" match the regular expression "`r*pa+|ps`"?

55/1p. Does the string "`yhhkkkk`" match the regular expression "`(yh)?(k*)?`"?

56/1p. Does the string "`tt`" match the regular expression "`t*(t*)?`"?

57/1p. Does the string "`zzw`" match the regular expression "`tw+uz|z?|w`"?

58/1p. Does the empty string match the regular expression "`(wn)+(yz)+|(px)?`"?

59/1p. Does the string "`fgc`" match the regular expression "`((fg)*c)?|(bu)?`"?

60/1p. Does the string "`uwwwww`" match the regular expression "`(ug)?|(wq)?|ww?`"?

61/1p. Does the string "`urdxdxtnnwb`" match the regular expression "`ur?(dx)*tn+wb`"?

62/1p. Does the string "`gxxxx`" match the regular expression "`(sy)?|g(xx)+a?`"?

63/1p. Does the string "`ndfndfhr`" match the regular expression "`((n*df)*hr)+`"?

64/1p. Does the string "`wwdtttwwwdtttw`" match the regular expression "`((w?dt)?|tw)+`"?

65/1p. Does the string "`uffffsh`" match the regular expression "`zx|uf*f+sh`"?

66/1p. Does the string "`pzw`" match the regular expression "`x*|pz|((gc)*w)?`"?

67/1p. Does the string "`zwyfc`" match the regular expression "`(zw)?|(yt)*(fc)*`"?

68/1p. Does the string "`zzz`" match the regular expression "`(z+|d(kn)*|y?)?`"?

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) $n^2 \lg n$ (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]);
   if ( simpleCompare ) {
      if ( simpleCompare ) {
         d = n; do {
            if ( simpleCompare ) {
               simpleStatement;
            } else {
               simpleStatement;
            }
         d--; } while ( d > 1 );
      }
   } else {
      if ( simpleCompare ) {
         simpleStatement;
      }
   }
   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^3 (C) $n^2 \lg n$ (E) $n\sqrt{n} \lg n$ (G) n (I) $\sqrt{n}$
(B) $n^2\sqrt{n}$ (D) n^2 (F) $n\sqrt{n}$ (H) $\sqrt{n} \lg^2 n$ (J) 1

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

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

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   for ( e = n ; e > 1 ; e-- ) {
      for ( k = n ; k > 1 ; k-- ) {
         if ( simpleCompare ) {
            if ( simpleCompare ) {
               for ( h = n ; h > 1 ; h-- ) {
                  simpleStatement;
               }
            }
         } else {
            simpleStatement;
         }
      }
   }
   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) $n\sqrt{n}$ (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) $\lg^3 n$ (J) 1

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

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   if ( simpleCompare ) {
      for ( k = 1 ; k < n ; k *= 2 ) {
         if ( simpleCompare ) {
            f = n; while ( f > 1 ) {
               b = 1; do {
                  if ( simpleCompare ) {
                     simpleStatement;
                  } else {
                     simpleStatement;
                  }
               b++; } while ( b < n );
            f /= 2; }
         }
      }
   }
   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) $\sqrt{n} \lg n$ (I) $\lg n$
(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 ) {
      if ( simpleCompare ) {
         if ( simpleCompare ) {
            if ( simpleCompare ) {
               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^5 (C) $n^2 \lg^2 n$ (E) n^2 (G) $n \lg n$ (I) $\sqrt{n}$
(B) $n^4\sqrt{n}$ (D) $n^2 \lg n$ (F) $n\sqrt{n}\lg^2 n$ (H) $\sqrt{n}\lg^4 n$ (J) 1

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   if ( simpleCompare ) {
      for ( a = n ; a > 1 ; a /= 3 ) {
         for ( k = 1 ; k * k < n ; k++ ) {
            if ( simpleCompare ) {
               for ( h = n ; h > 1 ; h-- ) {
                  for ( d = 1 ; d * d < n ; d += 3 ) {
                     for ( b = n ; b > 1 ; b /= 5 ) {
                        if ( simpleCompare ) {
                           if ( simpleCompare ) {
                              if ( simpleCompare ) {
                                 simpleStatement;
                              }
                           } else {
                              simpleStatement;
                           }
                        } else {
                           simpleStatement;
                        }
                     }
                  }
               }
            } else {
               for ( c = 1 ; c < n ; c *= 2 ) {
                  for ( g = 1 ; g < n ; g *= 5 ) {
                     if ( simpleCompare ) {
                        if ( simpleCompare ) {
                           simpleStatement;
                        }
                     } else {
                        simpleStatement;
                     }
                  }
               }
            }
         }
      }
   }
   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^3 (C) $n^2 \lg n$ (E) $n\sqrt{n} \lg n$ (G) $n \lg n$ (I) $\lg n$
(B) $n^2\sqrt{n}$ (D) n^2 (F) $n\sqrt{n}$ (H) $\sqrt{n}$ (J) 1

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   if ( simpleCompare ) {
      a = n; do {
         if ( simpleCompare ) {
            if ( simpleCompare ) {
               c = 1; do {
                  for ( b = n ; b > 1 ; b /= 5 ) {
                     simpleStatement;
                  }
               c *= 3; } while ( c < n );
            } else {
               simpleStatement;
            }
         } else {
            for ( f = 1 ; f * f < n ; f += 1 ) {
               h = 1; while ( h < n ) {
                  simpleStatement;
               h *= 2; }
            }
         }
      a--; } while ( a > 1 );
   } else {
      j = 1; while ( j * j < n ) {
         if ( simpleCompare ) {
            d = 1; while ( d < n ) {
               if ( simpleCompare ) {
                  simpleStatement;
               } else {
                  simpleStatement;
               }
            d += 3; }
         } else {
            i = 1; do {
               simpleStatement;
            i += 1; } while ( i * i < n );
         }
      j++; }
   }
   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^7 (C) $n^4 \lg^3 n$ (E) $n^4 \lg n$ (G) $n^2\sqrt{n}\lg^4 n$ (I) $n\sqrt{n}\lg^4 n$
(B) $n^4\sqrt{n}\lg n$ (D) $n^4 \lg^2 n$ (F) $n^3 \lg^2 n$ (H) $n\sqrt{n}\lg^5 n$ (J) $\lg^2 n$

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   f = 1; while ( f * f < n ) {
      for ( g = n ; g > 1 ; g-- ) {
         for ( d = 1 ; d * d < n ; d++ ) {
            if ( simpleCompare ) {
               for ( h = 1 ; h < n ; h += 2 ) {
                  if ( simpleCompare ) {
                     for ( b = 1 ; b < n ; b += 10 ) {
                        for ( c = 1 ; c < n ; c *= 3 ) {
                           j = n; while ( j > 1 ) {
                              simpleStatement;
                           j /= 2; }
                        }
                     }
                  } else {
                     i = n; do {
                        simpleStatement;
                     i--; } while ( i > 1 );
                  }
               }
            } else {
               k = 1; while ( k * k < n ) {
                  if ( simpleCompare ) {
                     if ( simpleCompare ) {
                        if ( simpleCompare ) {
                           simpleStatement;
                        } else {
                           simpleStatement;
                        }
                     } else {
                        simpleStatement;
                     }
                  } else {
                     simpleStatement;
                  }
               k++; }
            }
         }
      }
   f += 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^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) $\lg n$

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

Total points 100.

Answer Key (points per line)

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

Total points 100.