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

1/2p. 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 (H) $\sqrt{n}$ (J) 1

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

2/2p. 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) $\lg^2 n$ (J) 1

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

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

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

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

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

5/2p. Give a tight big-oh Θ() bound on the running time T(n) of this program.
Assume `atoi`, `simpleStatement`, and `simpleCompare` each run in Θ(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]);
   for ( c = 1 ; c * c < n ; c += 1 ) {
      if ( simpleCompare ) {
         d = 1; while ( d * d < n ) {
            simpleStatement;
         d += 10; }
      } else {
         simpleStatement;
      }
   }
   return 0; }
```

6/2p. Give a tight big-oh Θ() bound on the running time T(n) of this program.
Assume `atoi`, `simpleStatement`, and `simpleCompare` each run in Θ(1) time.

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

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

7/2p. 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) $\sqrt{n}$ (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 ) {
      b = 1; do {
         if ( simpleCompare ) {
            simpleStatement;
         } else {
            simpleStatement;
         }
      b++; } while ( b < n );
   } else {
      simpleStatement;
   }
   return 0; }
```

8/2p. 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) $n\sqrt{n}$ (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 ) {
            if ( simpleCompare ) {
               simpleStatement;
            }
         } else {
            simpleStatement;
         }
      }
   } else {
      if ( simpleCompare ) {
         if ( simpleCompare ) {
            simpleStatement;
         } else {
            simpleStatement;
         }
      } else {
         simpleStatement;
      }
   }
   return 0; }
```

9/2p. 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 ( i = 1 ; i * i < n ; i++ ) {
               simpleStatement;
            }
         }
      } else {
         f = 1; while ( f < n ) {
            simpleStatement;
         f *= 5; }
      }
   }
   return 0; }
```

10/2p. 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) 1

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

11/2p. 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 ) {
         i = 1; while ( i < n ) {
            if ( simpleCompare ) {
               simpleStatement;
            } else {
               simpleStatement;
            }
         i *= 2; }
      }
   } else {
      g = 1; do {
         simpleStatement;
      g *= 2; } while ( g < n );
   }
   return 0; }
```

12/2p. 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 (I) $\lg^2 n$
(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 ) {
      if ( simpleCompare ) {
         if ( simpleCompare ) {
            if ( simpleCompare ) {
               for ( b = n ; b > 1 ; b-- ) {
                  simpleStatement;
               }
            }
         } else {
            simpleStatement;
         }
      } else {
         for ( f = 1 ; f * f < n ; f += 3 ) {
            simpleStatement;
         }
      }
   }
   return 0; }
```

13/2p. 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) $\sqrt{n} \lg^2 n$ (G) $\sqrt{n}$ (I) $\lg n$
(B) $n^2\sqrt{n}$ (D) n^2 (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 ) {
            j = 1; do {
               i = n; do {
                  for ( c = n ; c > 1 ; c -= 5 ) {
                     simpleStatement;
                  }
               i -= 3; } while ( i > 1 );
            j++; } while ( j * j < n );
         } else {
            if ( simpleCompare ) {
               if ( simpleCompare ) {
                  simpleStatement;
               }
            } else {
               simpleStatement;
            }
         }
      }
   } else {
      if ( simpleCompare ) {
         k = n; do {
            if ( simpleCompare ) {
               b = 1; while ( b < n ) {
                  simpleStatement;
               b++; }
            } else {
               simpleStatement;
            }
         k /= 5; } while ( k > 1 );
      } else {
         g = n; do {
            simpleStatement;
         g /= 2; } while ( g > 1 );
      }
   }
   return 0; }
```

14/2p. 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) $\sqrt{n}$ (I) $\lg^2 n$
(B) $n^2 \lg n$ (D) $n\sqrt{n}$ (F) $\sqrt{n} \lg n$ (H) $\lg^3 n$ (J) $\lg n$

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

15/2p. 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) $\sqrt{n}\lg^2 n$ (H) $\sqrt{n}$ (J) 1

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

Total points 30.

Answer Key (points per line)

1 (2).	H ($\sqrt{n}$)
2 (2).	G ($\sqrt{n}\lg n$)
3 (2).	D ($n\sqrt{n}\lg n$)
4 (2).	C ($n^3\lg n$)
5 (2).	E (n)
6 (2).	D ($n\sqrt{n}\lg n$)
7 (2).	F (n)
8 (2).	J (1)
9 (2).	G ($\sqrt{n}$)
10 (2).	C (n^2)
11 (2).	J ($\lg n$)
12 (2).	G (n)
13 (2).	B ($n^2\sqrt{n}$)
14 (2).	C (n^2)
15 (2).	C ($n\sqrt{n}\lg n$)

Total points 30.