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

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   for ( h = n ; h > 1 ; h /= 5 ) {
      g = 1; while ( g * g < n ) {
         e = 1; do {
            for ( k = 1 ; k * k < n ; k++ ) {
               if ( simpleCompare ) {
                  for ( f = 1 ; f * f < n ; f += 10 ) {
                     if ( simpleCompare ) {
                        simpleStatement;
                     }
                  }
               }
            }
         e += 5; } while ( e * e < n );
      g++; }
   }
   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) $\lg^2 n$
(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]);
   if ( simpleCompare ) {
      for ( e = n ; e > 1 ; e -= 5 ) {
         if ( simpleCompare ) {
            simpleStatement;
         } 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) $n^2 \lg n$ (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) 1

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

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

5/2p. 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) $n \lg^2 n$ (G) n (I) $\lg^2 n$
(B) $n^2 \lg n$ (D) $n\sqrt{n}$ (F) $n \lg n$ (H) $\sqrt{n} \lg^2 n$ (J) $\lg n$

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

6/2p. 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) $n \lg n$ (H) $\lg^3 n$ (J) 1

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

```
int main ( int argc, char * * argv ) {
   int n = atoi(argv[1]);
   if ( simpleCompare ) {
      e = n; do {
         if ( simpleCompare ) {
            if ( simpleCompare ) {
               if ( simpleCompare ) {
                  simpleStatement;
               } else {
                  simpleStatement;
               }
            } else {
               simpleStatement;
            }
         }
      e--; } while ( e > 1 );
   } else {
      if ( simpleCompare ) {
         for ( c = 1 ; c * c < n ; c += 5 ) {
            a = 1; do {
               simpleStatement;
            a++; } while ( a * a < n );
         }
      }
   }
   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) $n\sqrt{n}$ (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 ) {
         for ( f = 1 ; f * f < n ; f += 10 ) {
            simpleStatement;
         }
      } else {
         simpleStatement;
      }
   } else {
      if ( simpleCompare ) {
         simpleStatement;
      }
   }
   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) $n\sqrt{n} \lg n$ (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 ) {
      g = n; while ( g > 1 ) {
         if ( simpleCompare ) {
            d = 1; while ( d * d < n ) {
               simpleStatement;
            d++; }
         } else {
            simpleStatement;
         }
      g--; }
   } else {
      for ( b = n ; b > 1 ; b-- ) {
         if ( simpleCompare ) {
            simpleStatement;
         } else {
            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 (E) $n\sqrt{n}$ (G) $\sqrt{n}\lg^2 n$ (I) $\lg^3 n$
(B) $n^2\sqrt{n}$ (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 ) {
      for ( g = 1 ; g * g < n ; g += 10 ) {
         for ( h = 1 ; h < n ; h *= 5 ) {
            if ( simpleCompare ) {
               k = 1; while ( k < n ) {
                  if ( simpleCompare ) {
                     simpleStatement;
                  } else {
                     simpleStatement;
                  }
               k++; }
            } else {
               if ( simpleCompare ) {
                  simpleStatement;
               } else {
                  simpleStatement;
               }
            }
         }
      }
   } else {
      if ( simpleCompare ) {
         if ( simpleCompare ) {
            if ( simpleCompare ) {
               f = 1; do {
                  simpleStatement;
               f++; } while ( f * f < n );
            }
         }
      } else {
         if ( simpleCompare ) {
            simpleStatement;
         } else {
            simpleStatement;
         }
      }
   }
   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^4 (C) $n^3 \lg n$ (E) n^2 (G) $n\sqrt{n}$ (I) $\sqrt{n}\lg^3 n$
(B) $n^3\sqrt{n}$ (D) n^3 (F) $n\sqrt{n}\lg n$ (H) n (J) $\lg n$

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

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

Total points 30.

Answer Key (points per line)

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

Total points 30.