Exercise Running Time Analysis of Functions in C

For the following C functions, determine the asymptotic time complexity depending on n as upper bounds and as short as possible. (Necessary preprocessor directives using `#include` are omitted for clarity.)

a)
```c
void functionA(int n)
{
   for (int i = n; i > 1; i -= 2)
   {
         printf(i);
   }
}
```

b)
```c
void functionB(int n)
{
   for (int i = 1; i < n; i++)
   {
         for (int j = n; j > n - i; j--)
         {
               printf(j);
         }
   }
}
```

c)
```c
void functionC(int n)
{
   for (int i = n; i > 1; i = i / 2)
   {
         for (int j = 1; j <= n; j++)
         {
               printf("Hello World!");
         }
   }
}
```

d)
```c
int functionD(int n)
{
   int z = 0;
   for (int i = 0; i < n; i++)
   {
         for (int j = 0; j < n; j++)
         {
               for (int k = 0; k < j; k++)
               {
                     z = z + 1;
               }
         }
   }
   return z;
}
```

e)
```
void functionE(int n)
{
   for (int i = n; i <= n; i++)
   {
         for (int j = n; j > 1; j = j / 2)
         {
               printf(j);
         }
   }
}
```

f)
```
void functionF(int n)
{
   for (int i = 1; i <= n * n; i += 10)
   {
         for (int j = 1; j * j <= n; j++)
         {
               printf("Hello World!");
         }
   }
}
```

g)
```
int functionG(int n)
{
   int z = 42;
   for (int i = 1; i < n; i++)
   {
         for (int j = 1; j < n * n + 1; j++)
         {
               return z;
         }
   }
}
```

h)
```
int functionH(int a, int n)
{
   if (n == 1){
         return a;
   }
   else{
         return a + functionH(a, n/2);
   }
}
```