The problem is binary search (Exercise 187), but each iteration tests to see if the item in the middle of the remaining segment is the item we seek.

(a) Write the program, with specifications and proofs.

Let the list be \( L \) and the value we are looking for be \( x \) (these are not state variables). Our program will assign natural variable \( i \) the index of an occurrence of \( x \) in \( L \) if \( x \) is there. Let's indicate whether \( x \) is present in \( L \) by assigning binary variable \( p \) the value \( \top \) if it is and \( \bot \) if not. Ignoring time for the moment, the problem is

\[
\begin{align*}
& x : L (0,..#L) = p' \implies L i' = x
\end{align*}
\]

As the search progresses, we narrow the segment of the list that we need to search. Let us introduce natural variables \( h \) and \( j \), and let specification \( R \) describe the search within the segment \( h,..j \).

\[
\begin{align*}
& R = (x : L (h,..j) = p' \implies L i' = x)
\end{align*}
\]

We can now solve the problem.

\[
\begin{align*}
& (x : L(0,..#L) = p' \implies L i' = x) \iff h := 0. \ j := #L. \ h < j \implies R
\end{align*}
\]

\[
\begin{align*}
& h < j \implies R \iff i := \text{div} (h + j) 2. \\
& \hspace{1cm} \text{if } L i = x \text{ then } p := \top \\
& \hspace{1cm} \text{else if } L i < x \text{ then } h := i + 1 \ \text{else } j := i \ 	ext{fi.} \\
& h \leq j \implies R \ 	ext{fi}
\end{align*}
\]

\[
\begin{align*}
& h \leq j \implies R \iff \text{if } h = j \text{ then } p := \bot \ \text{else } h < j \implies R \ 	ext{fi}
\end{align*}
\]

Before proving these three refinements, here are two little theorems that will be useful; let's call the first one Lemma A:

\[
\begin{align*}
& h < j = \text{div} (h + j) 2 + 1 \leq j
\end{align*}
\]

Here is its proof:

\[
\begin{align*}
& \text{if even } (h + j) \\
& \hspace{1cm} \text{then } \text{div} (h + j) 2 + 1 \leq j = (h + j)/2 + 1 \leq j = h + 2 \leq j = h < j \quad \text{note below} \\
& \hspace{1cm} \text{else } \text{div} (h + j) 2 + 1 \leq j = (h + j - 1)/2 + 1 \leq j = h + 1 \leq j = h < j \ 	ext{fi}
\end{align*}
\]

The “note below” refers to the fact that when \( h + j \) is even, \( h + 2 \leq j = h < j \).

And of course we'll call the other little theorem Lemma B:

\[
\begin{align*}
& h < j \implies h \leq \text{div} (h + j) 2
\end{align*}
\]

Here is its proof:

\[
\begin{align*}
& \text{if even } (h + j) \\
& \hspace{1cm} \text{then } h \leq \text{div} (h + j) 2 = h \leq (h + j)/2 = h \leq j \iff h < j \\
& \hspace{1cm} \text{else } h \leq \text{div} (h + j) 2 = h \leq (h + j - 1)/2 = h + 1 \leq j = h < j \ 	ext{fi}
\end{align*}
\]

For the proof of the first refinement we start with the right side:

\[
\begin{align*}
& h := 0. \ j := #L. \ h < j \implies R \quad \text{replace } R \text{ and then use Substitution Law twice} \\
& \hspace{1cm} \iff (x : L(0,..#L) = p' \implies L i' = x) \quad \text{we are given that } L \text{ is nonempty} \\
& \hspace{1cm} \iff (x : L(0,..#L) = p' \implies L i' = x)
\end{align*}
\]

The proof of the middle refinement, starting with the right side:

\[
\begin{align*}
& i := \text{div} (h + j) 2.
\end{align*}
\]
if \( L \ i = x \) then \( p := \top \)
else if \( L \ i < x \) then \( h := i+1 \) else \( j := i \).
\[
\text{replace } p \text{ assignment and } R
\]

\[
= \quad i := \text{div} \ (h+j) \ 2.
\]

\[
\text{if } L \ i = x \text{ then } p' = \top \land h' = h \land i' = i \land j' = j
\]
else if \( L \ i < x \) then \( h := i+1 \) else \( j := i \).
\[
\text{distribute final line back}
\]

\[
= \quad i := \text{div} \ (h+j) \ 2.
\]

\[
\text{if } L \ i = x \text{ then } p' \land h' = h \land i' = i \land j' = j
\]
else if \( L \ i < x \) then \( i+1 \leq j \Rightarrow (x: L(h_{..j}) = p' \Rightarrow L \ i' = x) \)
else \( j := i \).
\[
\text{substitution law twice on final two lines}
\]

\[
= \quad \text{if } L(\text{div} \ (h+j) \ 2) = x \text{ then } p' \land h' = h \land i' = \text{div} \ (h+j) \ 2 \land j' = j
\]
else if \( L(\text{div} \ (h+j) \ 2) < x \)
\[
\text{substitution law again}
\]

\[
= \quad L(\text{div} \ (h+j) \ 2) = x \land p' \land h' = h \land i' = \text{div} \ (h+j) \ 2 \land j' = j
\]
\[
\land \quad (\text{div} \ (h+j) \ 2 + 1 \leq j \Rightarrow (x: L(\text{div} \ (h+j) \ 2 + 1_{..j}) = p' \Rightarrow L \ i' = x))
\]
\[
\lor \quad \text{L(div} \ (h+j) \ 2/photo) < x
\]
\[
\land \quad (h \leq \text{div} \ (h+j) \ 2 \Rightarrow (x: L(h_{..\text{div} \ (h+j) \ 2}) = p' \Rightarrow L \ i' = x))
\]

\[
\text{case analysis}
\]

We must prove that this disjunction implies the left side of the refinement
\[
h < j \Rightarrow R
\]
By an antidistributive law, that is equivalent to proving that each of the three disjuncts
implies the left side of the refinement. Since the left side of the refinement is an
implication, by portation, we prove that its antecedent \( h < j \), and a disjunct, implies its
consequent \( R \), which is \( x: L(h_{..j}) = p' \Rightarrow L \ i' = x \). So, \( h < j \) and first disjunct:

\[
h < j \land L(\text{div} \ (h+j) \ 2) = x \land p' \land h' = h \land i' = \text{div} \ (h+j) \ 2 \land j' = j
\]
\[
\Rightarrow (x: L(h_{..j}) = p' \Rightarrow L \ i' = x)
\]

Middle disjunct:
\[
h < j \land L(\text{div} \ (h+j) \ 2) < x
\]
\[
\land \quad (\text{div} \ (h+j) \ 2 + 1 \leq j \Rightarrow (x: L(\text{div} \ (h+j) \ 2 + 1_{..j}) = p' \Rightarrow L \ i' = x))
\]
\[
\text{Use Lemma A for discharge}
\]

\[
= \quad h < j \land L(\text{div} \ (h+j) \ 2) < x \land (x: L(\text{div} \ (h+j) \ 2 + 1_{..j}) = p' \Rightarrow L \ i' = x)
\]
If \( h < j \) and \( L(\text{div} \ (h+j) \ 2) < x \) and \( L \) is sorted, then
\[
x: L(\text{div} \ (h+j) \ 2 + 1_{..j}) = x: L(h_{..j})
\]
\[
\Rightarrow (x: L(h_{..j}) = p' \Rightarrow L \ i' = x)
\]

specialize

\[
= \quad h < j \land L(\text{div} \ (h+j) \ 2) < x \land (x: L(h_{..j}) = p' \Rightarrow L \ i' = x)
\]
\[
\Rightarrow (x: L(h_{..j}) = p' \Rightarrow L \ i' = x)
\]
For recursive execution time, put \( t := t + 1 \) before the final, recursive call. We prove

\[
T \iff h := 0. \ j := \#L. \ U = i := \text{div} (h+j) 2. \\
\begin{array}{l}
\text{if } L \ i = x \text{ then } p := \top \\
\text{else if } L \ i < x \text{ then } h := i + 1 \text{ else } j := i. \\
V \ f i
\end{array}
\]

\[
V \iff \text{if } h=j \text{ then } p := \bot \text{ else } t := t + 1. \ U \ f i
\]

I know it's \( \log \) time, but to get a good upper bound I tried a few list sizes by hand, with the value sought \( x \) less than all list values. So \( T, U, \) and \( V \) are defined as

\[
T = t' \leq t + \text{floor} (\log (\#L)) \\
U = h < j \Rightarrow t' \leq t + \text{floor} (\log (j-h)) \\
V = \text{if } h = j \text{ then } t' = t \text{ else } t := t + 1. \ U \ f i
\]

where \( \text{floor} \) is the function that rounds down.

Before proving the refinements, here's another useful theorem, which I'll call Lemma C.

\[
1 + \text{floor} (\log (j - \text{div} (h+j) 2 - 1)) \leq \text{floor} (\log (j-h)).
\]

Here's the proof.

\[
\begin{array}{l}
\text{if } \text{even} (h+j) \\
\text{then} \quad 1 + \text{floor} (\log (j - \text{div} (h+j) 2 - 1)) \\
\quad = \text{floor} (1 + \log (j - (h+j)/2 - 1)) \\
\quad = \text{floor} (\log (j-h-2)) \\
\quad \leq \text{floor} (\log (j-h)) \\
\text{else} \quad 1 + \text{floor} (\log (j - \text{div} (h+j) 2 - 1)) \\
\quad = \text{floor} (1 + \log (j - (h+j-1)/2 - 1)) \\
\quad = \text{floor} (\log (j-h-1)) \\
\quad \leq \text{floor} (\log (j-h)) \ f i
\end{array}
\]

The first refinement is

\[
(T \iff h := 0. \ j := \#L. \ U) \quad \text{replace } T \text{ and } U = (t' \leq t + \text{floor} (\log (\#L)) \iff h := 0. \ j := \#L. \ h < j \Rightarrow t' \leq t + \text{floor} (\log (j-h)))
\]

Substitution Law twice
The middle refinement, starting with the right side, is

\[
\begin{align*}
&= i := \text{div} (h+j) 2. \\
&\quad \text{if } L i = x \text{ then } p := \top \\
&\quad \text{else if } L i < x \text{ then } h := i+1 \text{ else } j := i. \\
&\quad V \text{ fi} \\
&\text{distribute last line back}
\end{align*}
\]

=  

\[
\begin{align*}
&= i := \text{div} (h+j) 2. \\
&\quad \text{if } L i = x \text{ then } p' \land h' = h \land i' = i \land j' = j \land t' = t \\
&\quad \text{else if } L i < x \text{ then } h := i+1. \\
&\quad V \text{ else } j := i. \text{ V fi} \text{ fi replace two Vs and assignment to } p
\end{align*}
\]

=  

\[
\begin{align*}
&= i := \text{div} (h+j) 2. \\
&\quad \text{if } L i = x \text{ then } p' \land h' = h \land i' = i \land j' = j \land t' = t \\
&\quad \text{else if } L i < x \text{ then } h := i+1. \\
&\quad \text{if } h=j \text{ then } t' = t \text{ else } h < j \Rightarrow t' \leq t + 1 + \text{floor} (\log(j-h)) \text{ fi} \\
&\quad \text{else } j := i. \text{ if } h=j \text{ then } t' = t \text{ else } h < j \Rightarrow t' \leq t + 1 + \text{floor} (\log(j-h)) \text{ fi} \text{ fi} \text{ fi substitution law for } h \text{ and } j
\end{align*}
\]

=  

\[
\begin{align*}
&= \text{if } L(\text{div} (h+j) 2) = x \text{ then } p' \land h' = h \land i' = \text{div} (h+j) 2 \land j' = j \land t' = t \\
&\quad \text{else if } L(\text{div} (h+j) 2) < x \text{ then } h := \text{div} (h+j) 2 + 1 = j \land t' = t \\
&\quad \text{else if } h = \text{div} (h+j) 2 \text{ then } t' = t \\
&\quad \text{else } h < \text{div} (h+j) 2 \Rightarrow t' \leq t + 1 + \text{floor} (\log(h+j) 2 - h) \text{ fi} \text{ fi} \text{ fi} \text{ case analysis law}
\end{align*}
\]

=  

\[
\begin{align*}
&= L(\text{div}(h+j)2) = x \land p' \land h' = h \land i' = \text{div} (h+j) 2 \land j' = j \land t' = t \\
&\lor L(\text{div}(h+j)2) < x \land \text{div}(h+j)2 + 1 = j \land t' = t \\
&\land (\text{div}(h+j)2 + 1 < j \Rightarrow t' \leq t + 1 + \text{floor} (\log(j - \text{div}(h+j)2 - 1))) \\
&\lor L(\text{div}(h+j)2) > x \land h = \text{div}(h+j)2 \land t' = t \\
&\lor L(\text{div}(h+j)2) > x \land h > \text{div}(h+j)2 \\
&\land (h < \text{div} (h+j) 2 \Rightarrow t' \leq t + 1 + \text{floor} (\log(h+j) 2 - h))
\end{align*}
\]

We must prove that this disjunction implies the left side of the refinement \( U \). By an antidistributive law, that is equivalent to proving that each of the three disjuncts implies the left side of the refinement. Since the left side of the refinement

\[ h \land t' \leq t + \text{floor} (\log (j-h)) \]

is an implication, by portation, we prove that its antecedent \( h \land j \), and a disjunct, implies its consequent \( t' \leq t + \text{floor} (\log (j-h)) \). So, \( h \land j \) and first disjunct:

\[
\begin{align*}
&= h \land j \land t' = t \\
&\text{specialize arithmetic}
\end{align*}
\]
\[ t' \leq t + \text{floor} \left( \log (j-h) \right) \]

Second disjunct:

\[ h<j \land L(\text{div}(h+j)2) < x \land \text{div}(h+j)2 + 1 = j \land t'=t \]

\[ \Rightarrow \ h<j \land t'=t \]

\[ \Rightarrow \ t' \leq t + \text{floor} \left( \log (j-h) \right) \]

Next disjunct:

\[ h<j \land L(\text{div}(h+j)2) < x \land \text{div}(h+j)2 + 1 \neq j \]

\[ \Rightarrow \ h<j \land \text{div}(h+j)2 + 1 = j \]

\[ \Rightarrow \ t' \leq t+1 + \text{floor}(\log(j - \text{div}(h+j)2 - 1)) \]

That completes the proof of the middle refinement.

Finally, we have to prove the last refinement.

\[ (V \Leftarrow \text{if } h=j \text{ then } p:= \bot \text{ else } t:= t+1. \ U \text{ fi}) \]

Replace \( V \)

\[ = \ (\text{if } h=j \text{ then } t'=t \text{ else } t:= t+1. \ U \text{ fi}) \]

\[ = \ \top \]

(c) Find the execution time according to a measure that charges time 1 for each test.
\[ A \leftarrow h := 0, j := \#L, B \]

\[ B \leftarrow i := \text{div}(h+j) 2, t := t+1. \quad \text{this time increment is for the test } \text{if } L \ i = x \text{ then } p := \top \]

\[ \quad \text{else } t := t+1. \quad \text{this time increment is for the test } \text{if } L \ i < x \text{ then } h := i+1 \text{ else } j := i \text{ fi} \]

\[ C \leftarrow t := t+1. \quad \text{this time increment is for the test } \text{if } h < j \]

\[ \quad \text{if } h = j \text{ then } t' := t \text{ else } B \text{ fi} \]

I know it’s \( \log \) time, but to get a good upper bound I tried a few list sizes by hand, with the value sought \( x \) less than all list values. So \( A \), \( B \), and \( C \) are defined as

\[ A = t' \leq t + 3 + 3 \times \text{floor} (\log (\#L)) \]

\[ B = h < j \Rightarrow t' \leq t + 3 + 3 \times \text{floor} (\log (j-h)) \]

\[ C = t := t+1. \text{ if } h = j \text{ then } t' := t \text{ else } B \text{ fi} \]

\[ = \text{ if } t := t+1. \text{ h = j then } t := t+1. \text{ t' := t else } t := t+1. \text{ B fi} \]

\[ = \text{ if } h = j \text{ then } t' \leq t+1 \text{ else } h < j \Rightarrow t' \leq t + 4 + 3 \times \text{floor} (\log (j-h)) \text{ fi} \]

where \( \text{floor} \) is the function that rounds down.

These three refinements should now be proven, but I haven’t got the energy.

(d) Compare the execution time to binary search without the test for equality each iteration.

\[ A \]

The execution time bounds are:

- Ex.187 recursive time, without equality test \( \text{ceil} (\log (\#L)) \)
- Ex.188(b) recursive time, with equality test \( \text{floor} (\log (\#L)) \)
- Ex.188(c) counting tests, with equality test \( 3 + 3 \times \text{floor} (\log (\#L)) \)

Ex.187 and Ex.188(b) are almost the same; no significant difference.
Ex.188(c) takes about 3 times as long as Ex.187 and Ex.188(b).