1. (i.)

$$L(\lambda) = \prod_{i=1}^{r} \lambda e^{-\lambda x_i} \prod_{i=r+1}^{n} \mathbb{P}(X_i \leq c_i)$$

which simplifies to the expression given in the question.

(ii.) a) The easiest way is to note that the distribution of $X|X > z_i$ is still exponential($\lambda$) by the memoryless property, and so

$$\mathbb{E}(X|X > z_i) = z_i + \frac{1}{\lambda}$$

Alternatively, you can do this by deriving the pdf and then calculating the integral, which is fine as well, but takes more work.

b)

$$\begin{aligned} \mathbb{E}X &= \mathbb{E}(X|X < c_i)\mathbb{P}(X < c_i) + \mathbb{E}(X|X \geq c_i)\mathbb{P}(X \geq c_i) \\ \frac{1}{\lambda} &= \mathbb{E}(X_i|X_i < c_i)(1 - e^{-\lambda c_i}) + (c_i + \frac{1}{\lambda})e^{-\lambda c_i} \end{aligned}$$

Thus

$$L_i = \mathbb{E}(X_i|X_i \leq c_i) = \frac{1 - (1 + \lambda c_i)e^{-\lambda c_i}}{\lambda(1 - e^{-\lambda c_i})}$$

c) Introduce $X_{r+1}, \ldots, X_n$ as the missing data. Then

$$\begin{aligned} L(\lambda|x_{obs}, X_{mis}, \{c_i\}) &= \prod \lambda e^{-\lambda X_i} \\ &= \lambda^n e^{-\lambda \sum X_i} \end{aligned}$$

Thus

$$\begin{aligned} Q(\lambda, \lambda^{(m)}) &= \mathbb{E}_{X_{mis}|\lambda^{(m)}, x_{obs}, c}(\log L(\lambda|x_{obs}, X_{mis}, \{c_i\})) \\ &= n \log \lambda - \lambda \sum \mathbb{E}_{X|\lambda^{(m)}, x_{obs}, c} X_i \\ &= n \log \lambda - \lambda \sum_{i=1}^{r} x_i - \lambda \sum_{i=r+1}^{n} L_i^{(m)} \end{aligned}$$

where $L_i^{(m)}$ is an estimate of $\mathbb{E}(X_i|X_i \leq z_i)$ calculated using $\lambda^{(m)}$. This is minimized at

$$\hat{\lambda} = \lambda^{(m+1)} = \frac{n}{\sum_{i=1}^{r} x_i + \sum_{i=r+1}^{n} L_i^{(m)}}$$

The EM algorithm then iterates from some starting value of $\lambda^{(0)}$

- Calculate $L_i^{(m)}$ for $i = r + 1, \ldots, n$ given $\lambda^{(m)}$
- Calculate $\lambda^{(m+1)}$ given $L_i^{(m)}$ etc

d)

```
> n <- 1000
> lambda.true = 1
> X <- rexp(n,lambda.true) # generate some exponential random variables
> cc=1 # the threshold we left-censor at.
```

```
> Xobs = X[X>=cc] ## the X values we observe
> rr = length(Xobs)
> lambda <- c()
> lambda[1] <-0.5  # starting value for lambda. You should try a few values
> for(m in 1:20){
+   L <- (1-(1+lambda[m]*cc)*exp(-lambda[m]*cc))/
(lambda[m]*(1-exp(-lambda[m]*cc)))
+   lambda[m+1] = n/(sum(Xobs) + L*(n-rr))
+ }
> lambda
 [1] 0.5000000 0.9791505 1.0038032 1.0050789 1.0051449 1.0051483 1.0051485 1.00
 [9] 1.0051485 1.0051485 1.0051485 1.0051485 1.0051485 1.0051485 1.0051485 1.00
[17] 1.0051485 1.0051485 1.0051485 1.0051485 1.0051485
>
```

So you can see we quickly converge to the maximum likelihood estimator, which in this case is close to the true value of $\lambda$.

2.

```
> library(mice)
> md.pattern(mammalsleep)
   species bw brw pi sei odi ts mls gt ps sws
42       1  1   1  1   1   1  1   1  1  1   1  0
 2       1  1   1  1   1   1  1   0  1  1   1  1
 3       1  1   1  1   1   1  1   1  0  1   1  1
 9       1  1   1  1   1   1  1   1  1  0   0  2
 2       1  1   1  1   1   1  0   1  1  1   0  2
 1       1  1   1  1   1   1  1   0  0  1   1  2
 2       1  1   1  1   1   1  0   1  1  0   0  3
 1       1  1   1  1   1   1  1   0  1  0   0  3
         0  0   0  0   0   0  4   4  4 12  14 38
> md.pairs(mammalsleep)
$rr
        species bw brw sws ps ts mls gt pi sei odi
species      62 62  62  48 50 58  58 58 62  62  62
bw           62 62  62  48 50 58  58 58 62  62  62
brw          62 62  62  48 50 58  58 58 62  62  62
sws          48 48  48  48 48 48  45 44 48  48  48
ps           50 50  50  48 50 48  47 46 50  50  50
ts           58 58  58  48 48 58  54 54 58  58  58
mls          58 58  58  45 47 54  58 55 58  58  58
gt           58 58  58  44 46 54  55 58 58  58  58
pi           62 62  62  48 50 58  58 58 62  62  62
sei          62 62  62  48 50 58  58 58 62  62  62
odi          62 62  62  48 50 58  58 58 62  62  62

$rm
        species bw brw sws ps ts mls gt pi sei odi
species       0  0   0  14 12  4   4  4  0   0   0
bw            0  0   0  14 12  4   4  4  0   0   0
```

```
brw           0  0   0  14 12  4   4  4  0   0   0
sws           0  0   0   0  0  0   3  4  0   0   0
ps            0  0   0   2  0  2   3  4  0   0   0
ts            0  0   0  10 10  0   4  4  0   0   0
mls           0  0   0  13 11  4   0  3  0   0   0
gt            0  0   0  14 12  4   3  0  0   0   0
pi            0  0   0  14 12  4   4  4  0   0   0
sei           0  0   0  14 12  4   4  4  0   0   0
odi           0  0   0  14 12  4   4  4  0   0   0

$mr
        species bw brw sws ps ts mls gt pi sei odi
species       0  0   0   0  0  0   0  0  0   0   0
bw            0  0   0   0  0  0   0  0  0   0   0
brw           0  0   0   0  0  0   0  0  0   0   0
sws          14 14  14   0  2 10  13 14 14  14  14
ps           12 12  12   0  0 10  11 12 12  12  12
ts            4  4   4   0  2  0   4  4  4   4   4
mls           4  4   4   3  3  4   0  3  4   4   4
gt            4  4   4   4  4  4   3  0  4   4   4
pi            0  0   0   0  0  0   0  0  0   0   0
sei           0  0   0   0  0  0   0  0  0   0   0
odi           0  0   0   0  0  0   0  0  0   0   0

$mm
        species bw brw sws ps ts mls gt pi sei odi
species       0  0   0   0  0  0   0  0  0   0   0
bw            0  0   0   0  0  0   0  0  0   0   0
brw           0  0   0   0  0  0   0  0  0   0   0
sws           0  0   0  14 12  4   1  0  0   0   0
ps            0  0   0  12 12  2   1  0  0   0   0
ts            0  0   0   4  2  4   0  0  0   0   0
mls           0  0   0   1  1  0   4  1  0   0   0
gt            0  0   0   0  0  0   1  4  0   0   0
pi            0  0   0   0  0  0   0  0  0   0   0
sei           0  0   0   0  0  0   0  0  0   0   0
odi           0  0   0   0  0  0   0  0  0   0   0

> mammal.mice <- mice(mammalsleep, m=10, printFlag=F)
>
> (fit.mice <- with(mammal.mice, lm(ts ~ brw + bw)))
call :
with.mids(data = mammal.mice, expr = lm(ts ~ brw + bw))

call1 :
mice(data = mammalsleep, m = 10, printFlag = F)
```

```
nmis :
species      bw     brw     sws      ps      ts     mls      gt      pi     sei     odi
      0       0       0      14      12       4       4       4       0       0       0

analyses :
[[1]]

Call:
lm(formula = ts ~ brw + bw)

Coefficients:
(Intercept)          brw           bw
  11.201854    -0.002882     0.001190


[[2]]

Call:
lm(formula = ts ~ brw + bw)

Coefficients:
(Intercept)          brw           bw
  10.855112    -0.002890     0.001226


[[3]]

Call:
lm(formula = ts ~ brw + bw)

Coefficients:
(Intercept)          brw           bw
  11.093940    -0.003007     0.001296


[[4]]

Call:
lm(formula = ts ~ brw + bw)

Coefficients:
(Intercept)          brw           bw
  11.004127    -0.002968     0.001270


[[5]]
```

```
Call:
lm(formula = ts ~ brw + bw)

Coefficients:
(Intercept)          brw           bw
  11.044939    -0.003015     0.001302


[[6]]

Call:
lm(formula = ts ~ brw + bw)

Coefficients:
(Intercept)          brw           bw
  11.053669    -0.002389     0.000828


[[7]]

Call:
lm(formula = ts ~ brw + bw)

Coefficients:
(Intercept)          brw           bw
  10.767178    -0.002874     0.001217


[[8]]

Call:
lm(formula = ts ~ brw + bw)

Coefficients:
(Intercept)          brw           bw
  10.868251    -0.002916     0.001238


[[9]]

Call:
lm(formula = ts ~ brw + bw)

Coefficients:
(Intercept)          brw           bw
  11.079749    -0.002587     0.001011
```

```
[[10]]

Call:
lm(formula = ts ~ brw + bw)

Coefficients:
(Intercept)          brw           bw
 11.1348795   -0.0024429    0.0009024




>
> pool(fit.mice)
Call: pool(object = fit.mice)

Pooled coefficients:
 (Intercept)          brw           bw
11.010369978 -0.002797130  0.001147976

Fraction of information about the coefficients missing due to nonresponse:
(Intercept)         brw          bw
 0.08828243  0.05330465  0.04331295
> summary(pool(fit.mice))
                     est          se          t       df  Pr(>|t|)        lo 95
(Intercept) 11.010369978 0.618178095 17.8109999 53.03662 0.0000000  9.770481726
brw         -0.002797130 0.001737704 -1.6096705 55.81741 0.1131115 -0.006278420
bw           0.001147976 0.001788789  0.6417613 56.48839 0.5236240 -0.002434717
                   hi 95 nmis        fmi     lambda
(Intercept) 1.225026e+01   NA 0.08828243 0.05453800
brw         6.841597e-04    0 0.05330465 0.01998052
bw          4.730668e-03    0 0.04331295 0.01003016
>
> ## By hand
> (coefs.mice <- sapply(fit.mice$analyses,function(fit) coef(fit)))
                    [,1]         [,2]         [,3]         [,4]         [,5]
(Intercept) 11.201854059 10.855111751 11.093940391 11.004127194 11.044939018
brw         -0.002881564 -0.002889855 -0.003007445 -0.002968282 -0.003015497
bw           0.001190180  0.001226169  0.001295690  0.001269802  0.001302134
                     [,6]         [,7]         [,8]         [,9]         [,10]
(Intercept) 11.0536693569 10.767178204 10.868250805 11.079749494 11.1348795117
brw         -0.0023887106 -0.002874465 -0.002915976 -0.002586572 -0.0024429345
bw           0.0008279681  0.001216572  0.001237846  0.001011040  0.0009023565
> mean(coefs.mice[3,]) # gives the posterior expectation
[1] 0.001147976
>
> apply(coefs.mice,1,var) # gives the variance and bw estimates
```

```
 (Intercept)          brw           bw
1.894671e-02 5.484858e-08 2.917653e-08
>
> sapply(fit.mice$analyses, function(fit) vcov(fit)[3,3])
 [1] 3.088229e-06 3.057972e-06 3.163514e-06 3.292173e-06 3.207928e-06 3.174417e-06
 [7] 3.269411e-06 3.301737e-06 3.085114e-06 3.036241e-06
> # gives the 10 variance estimates
>
>
> # Lets combine these to get the posterior variance
> post.var <- mean(sapply(fit.mice$analyses, function(fit) vcov(fit)[3,3])) + (1+1/10)*
>
> # finally take the square root to get the standard error
> sqrt(post.var)
[1] 0.001788789
>
> # which agrees with the output from the pool command
>
```