1. Write a `pois.prob()` function that computes $P(X = x)$, $P(X \neq x)$, $P(X < x)$, $P(X \leq x)$, $P(X > x)$, and $P(X \geq x)$. Enable the user to specify the rate parameter $\lambda$.

```r
  pois.prob <- function(x, lambda, type= "=="){
  if(type == "=="){
    return(dpois(x, lambda)) #P(X=x) we use PMF
  }else if(type == "!="){
    return(1-dpois(x, lambda)) #P(X!=x), compliment rule with PMF
  }else if(type == "<"){
    return(ppois(x-1, lambda)) #P(X<x), CDF at x-1
  }else if(type == "<="){
    return(ppois(x, lambda)) #P(X<=x), CDF at x
  }else if(type == ">"){
    return(1-ppois(x, lambda)) #P(X<=x), compliment of CDF at x
  }else if(type == ">="){
    return(1-ppois(x-1, lambda)) #P(X<=x), CDF at x-1
  }
}
```

   Since the Poisson Distribution is a discrete distribution we use the `ppois()` function for the CDF and `dpois()` for the PDF. We use a combination of the CDF and PDF to compute each case.

2. Write a `beta.prob()` function that computes $P(X = x)$, $P(X \neq x)$, $P(X < x)$, $P(X \leq x)$, $P(X > x)$, and $P(X \geq x)$ for a beta distribution. Enable the user to specify the shape parameters $\alpha$ and $\beta$.

```r
beta.prob <- function(x, alpha, beta, type= "=="){
  if(type == "=="){
    return(0) #P(X=x) not possible in continuous case
  }else if(type == "!="){
    return(1) #P(X!=x), compliment rule from above
  }else if(type == "<"){
    return(pbeta(x, alpha, beta)) #P(X<x) CDF at x
  }else if(type == "<="){
    return(pbeta(x, alpha, beta)) #P(X<x)=P(X<=x) so same as above
  }else if(type == ">"){
    return(1-pbeta(x, alpha, beta)) #P(X>x) compliment of P(X<x)
  }else if(type == ">="){
    return(1-pbeta(x, alpha, beta)) #P(X>x)=P(X>=x) so same as above
  }
}
```

   Since the Beta Distribution is a continuous distribution we use the `pbeta()` function for the CDF. We could also use the `debta()` function, however, I did not find this necessary for computing the probabilities for each of these cases as using the CDF works just fine.