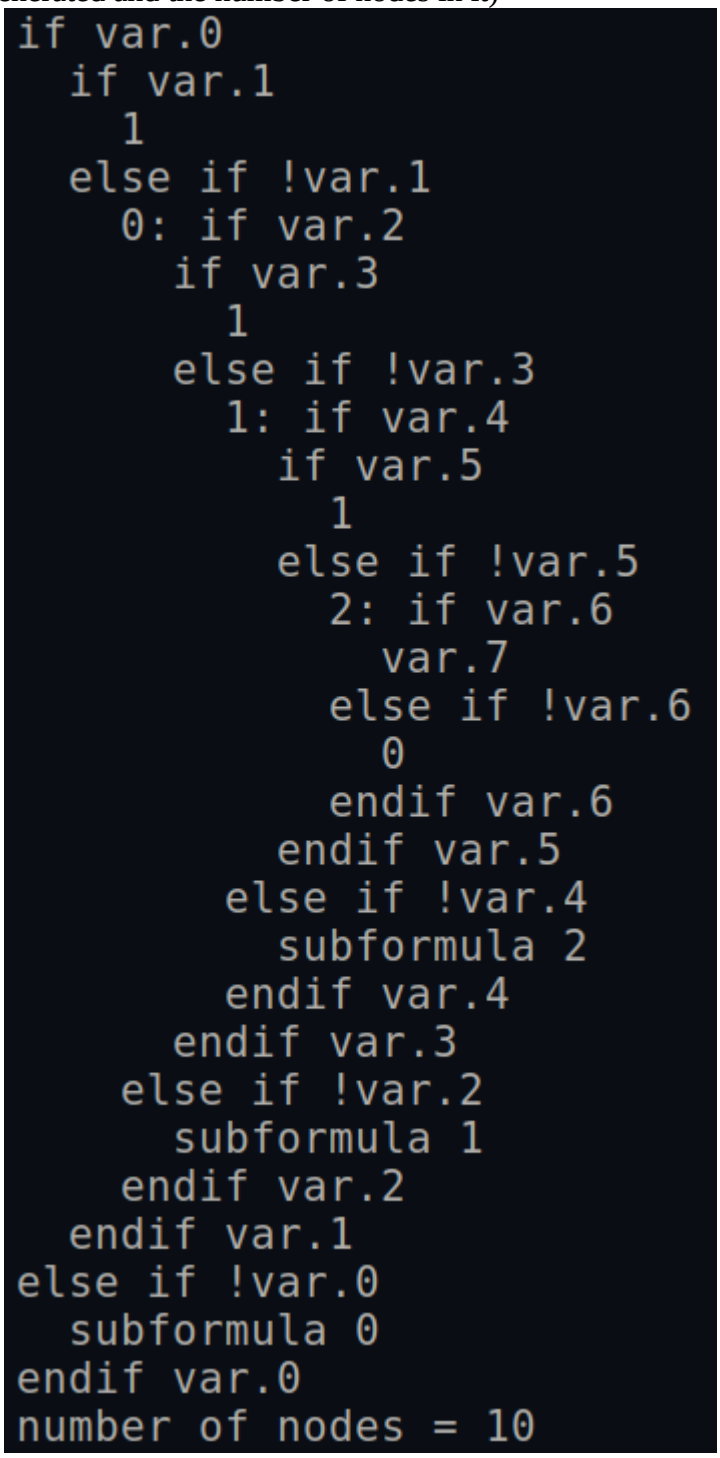
BDD Assignment, Sheel Shah, 19D070052, EE709
Note: since the outputs are long, some questions have multiple images which are to read one after the other to see the output.

Q1.
(the output shows the bdd generated and the number of nodes in it)



Q2.
(the output shows the bdd generated and the number of nodes in it)

```
if var.0
  if var.1
    if var.2
      if var.3
        if var.4
          1
        else if !var.4
          0: if var.5
            1
          else if !var.5
            1: if var.6
              1
            else if !var.6
              var.7
            endif var.6
          endif var.5
        endif var.4
      else if !var.3
        if var.4
          1
        else if !var.4
          2: if var.5
            1
          else if !var.5
            var.6
          endif var.5
        endif var.4
      endif var.3
    else if !var.2
      if var.3
        if var.4
          1
        else if !var.4
          3: if var.5
            1
          else if !var.5
            var.7
          endif var.5
        endif var.4
      else if !var.3
        if var.4
          1
        else if !var.4
          var.5
        endif var.4
      endif var.3
    endif var.2
  else if !var.1
    if var.2
      if var.3
        if var.4
          1
        else if !var.4
          subformula 1
        endif var.4
      else if !var.3
        if var.4
          1
        else if !var.4
          var.6
        endif var.4
      endif var.3
    else if !var.2
      if var.3
        if var.4
          1
        else if !var.4
          var.7
        endif var.4
      else if !var.3
        var.4
      endif var.3
    endif var.2
  endif var.1
else if !var.0
  if var.1
    if var.2
      if var.3
        subformula 0
      else if !var.3
```

```
        subformula 0
      else if !var.3
        subformula 2
      endif var.3
    else if !var.2
      if var.3
        subformula 3
      else if !var.3
        var.5
      endif var.3
    endif var.2
  else if !var.1
    if var.2
      if var.3
        subformula 1
      else if !var.3
        var.6
      endif var.3
    else if !var.2
      if var.3
        var.7
      else if !var.3
        0
      endif var.3
    endif var.2
  endif var.1
endif var.0
number of nodes = 32
```

Q3.
The bdd's are calculated using the ripple carry adder method.
For parts b and c, appropriate image and pre-image computation using the method taught in class is done. The code contains exact functions used.

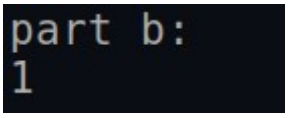
```
s1:
if var.0
  0: if var.1
    if var.2
      if var.3
        if var.4
          1
        else if !var.4
          1: if var.6
            0
          else if !var.6
            var.7
          endif var.6
        endif var.4
      else if !var.3
        !var.4
      endif var.3
    else if !var.2
      if var.3
        if var.4
          1
        else if !var.4
          2: if var.6
            var.7
          else if !var.6
            !var.7
          endif var.6
        endif var.4
      else if !var.3
        if var.4
          0
        else if !var.4
          var.6
        endif var.4
      endif var.3
    endif var.2
  else if !var.1
    if var.2
      if var.3
        if var.4
```

```
          subformula 1
        else if !var.4
          !subformula 1
        endif var.4
      else if !var.3
        var.4
      endif var.3
    else if !var.2
      if var.3
        if var.4
          subformula 2
        else if !var.4
          !subformula 2
        endif var.4
      else if !var.3
        if var.4
          var.6
        else if !var.4
          !var.6
        endif var.4
      endif var.3
    endif var.2
  endif var.1
else if !var.0
  !subformula 0
endif var.0
```

```
s2:
if var.1
  0: if var.2
    if var.3
      if var.4
        1: if var.6
          1
        else if !var.6
          !var.7
        endif var.6
      else if !var.4
        !subformula 1
      endif var.4
    else if !var.3
      var.4
    endif var.3
  else if !var.2
    if var.3
      if var.4
        2: if var.6
          !var.7
        else if !var.6
          var.7
        endif var.6
      else if !var.4
        !subformula 2
      endif var.4
    else if !var.3
      if var.4
        var.6
      else if !var.4
        !var.6
      endif var.4
    endif var.3
  endif var.2
else if !var.1
  !subformula 0
endif var.1
```

```
s3:
if var.2
  0: if var.3
    if var.6
      var.7
    else if !var.6
      !var.7
    endif var.6
  else if !var.3
    !var.6
  endif var.3
else if !var.2
  !subformula 0
endif var.2
```

```
s4:
if var.3
  !var.7
else if !var.3
  var.7
endif var.3
```




Output 1 means the image of the given is set is 1. Therefore, all possible outputs can be reached even after the constraints on the input.

```
part c:
if var.0
  0: if var.1
    if var.2
      if var.3
        if var.4
          if var.6
            1
          else if !var.6
            var.7
          endif var.6
        else if !var.4
          var.6
        endif var.4
      else if !var.3
        1: if var.6
          !var.7
        else if !var.6
          var.7
        endif var.6
      endif var.3
    else if !var.2
      if var.3
        if var.4
          var.7
        else if !var.4
          !var.6
        endif var.4
      else if !var.3
        if var.4
          var.7
        else if !var.4
          !subformula 1
        endif var.4
      endif var.3
    endif var.2
  else if !var.1
    if var.2
      2: if var.3
        var.6
```

```
        else if !var.4
          !subformula 1
        endif var.4
      endif var.3
    endif var.2
  else if !var.1
    if var.2
      2: if var.3
        var.6
      else if !var.3
        subformula 1
      endif var.3
    else if !var.2
      !subformula 2
    endif var.2
  endif var.1
else if !var.0
  !subformula 0
endif var.0
```

Q4.
We can easily check if a function is invertible if the existential quatification of the characteristic function with respect to the input variables is 1. This is because there are n input and n output variables, and hence one-oneness guarantees ontoness. And we can check one-oneness by simply seeing if all output combinations are reachable.

We test this using f1 as an invertible function vector, and f2 as a non-invertible one. (See code for exact details). The output is as follows: