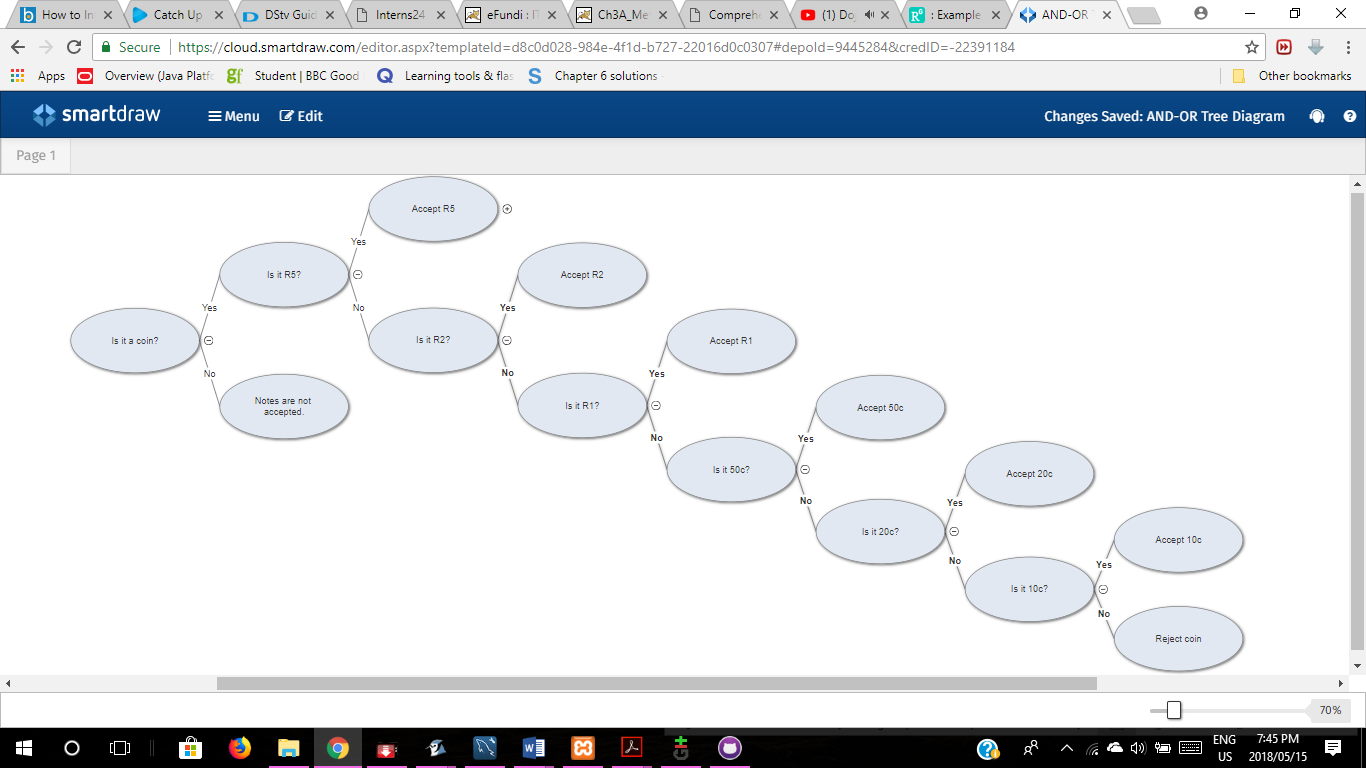
Planning FSM:

|  |  |  |
| --- | --- | --- |
| NAME | CONTRIBUTION DESCRIPTION | CONTRIBUTION SCORE |
| JOHN MOKOENA |  | 100% |
| KATLEHO MOKOENA |  | 100% |
| MAKGOTSO MOKOENA | End user documentation | 100% |
| T MOKOKA | Planning Doc | 100% |
| THAPELO MOLETSANE | Coding | 100% |
| KG MOOKETSI | Diagrams | 100% |
| MPHO MORIPE | Planning Doc | 100% |
| MATSHIDISO MOSHOALIBA | Planning Doc & End user Documentation | 100% |
| Y MOOSA | Coding | 100% |
| PAPI MORALLANE | Planning Doc | 100% |

Planning for the FSM:

Getting started was the hardest part due to nothing else but pure procrastination and hectic workloads but when we finally met we had the greatest mind shackle was how we’d create it but after the tree diagram was sorted we had a great sense of directions. We then distributed the work evenly and result in some of the following work:

Concluded FSM:

|  |  |
| --- | --- |
|  |  |

Project Goals:

The objective of this code is to draw up a FSM that will take in specific coins and modified to return change. In order to get the desired product the coin should meet the combinations of the states. This machine validates for other conditions, should the coins specified “ R5, R2, R1, c50, c20, c10” are not inserted it will prompt the user to enter the correct coins.

Implementation:

The parts of this code are explained in the order that makes the most sense to a person. Please look at the code if you want to see how it’s actually assembled.

; template to create items

(deftemplate items

(slot itemName)

(slot itemPrice)

)

; Facts used when instantiatng

(deffacts itemDetails

(items (itemName chocolate) (itemPrice 15.0))

(items (itemName cola) (itemPrice 8.5))

(items (itemName orange) (itemPrice 10.0))

(items (itemName sweets) (itemPrice 12.5))

)

; Function to retrieve amount of change if any

(deffunction paymentChange (?amountChange)

(if (< ?amountChange 0) then

(printout t "Dont forget your change: R " (\* ?amountChange -1) crlf ; used -1 because it retrieves a negative if there is change when we want positive

"Come back anytime for more goodies" crlf)

else

(printout t "You do not have any change, have a nice day" crlf)

)

)

; Function to calcuate and add a payment to be regstered

(deffunction addPayment (?calcAmount)

(while (> ?calcAmount 0)

(bind ?inputCoin (read)

)

(if(eq ?inputCoin R5) then

(bind ?calcAmount (- ?calcAmount 5.00)) ; Probably hould have used a facts for this

)

(if(eq ?inputCoin R2) then

(bind ?calcAmount (- ?calcAmount 2.00))

)

(if(eq ?inputCoin R1) then

(bind ?calcAmount (- ?calcAmount 1.00))

)

(if(eq ?inputCoin c50) then

(bind ?calcAmount (- ?calcAmount 0.50))

)

(if(eq ?inputCoin c20) then

(bind ?calcAmount (- ?calcAmount 0.20))

)

(if(eq ?inputCoin c10) then

(bind ?calcAmount (- ?calcAmount 0.10))

)

(if(> ?calcAmount 0) then

(printout t "You need R " ?calcAmount crlf))

)

(paymentChange ?calcAmount)

)

; Function to instantiate an item to be processed

(deffunction addItem ()

(bind ?choice (read))

(if (eq ?choice ch) then

(bind ?priceChoice 15.00)

)

(if (eq ?choice co) then

(bind ?priceChoice 8.50)

)

(if (eq ?choice or) then

(bind ?priceChoice 10.00)

)

(if (eq ?choice sw) then

(bind ?priceChoice 12.50)

)

(printout t "You need R " ?priceChoice crlf)

(printout t "You may enter the following amounts: crlf

R5, R2, R1, c50, c20, c10" crlf

"Enter amount for input : " )

(addPayment ?priceChoice)

)

; Rule used to as the user what item they would like

(defrule print

=>

(printout t "Choose one of the following : " crlf

"Chocolate : ch" crlf

"Cola : co" crlf

"Orange : or" crlf

"Sweets : sw" crlf crlf

(printout t "Your choice: ")

(addItem)

)

*Default State*

So in the default state of the program, all the items together with their prices are shown to the user (See deffacts code). Show the user decide that they want to buy a certain item, the program will then move to the follow state.

*Follow State*

The follow state allows the user to enter the name of the item they want and how much they will be paying. After entering the amount they will be paying, the program will then calculate and give back the output of the change, if any. The program will then process the item and then add the payment to the register.

Communication between group members:

