**Motivation**

1. **SHACS Speaker who graduated over 24 years ago now crafting software to pump oil from wells, especially wells that are no longer productive: The most important thing I learned at SHSU was the use of finite state automata in the Language Translator course for building machines to solve the data parsing problem.**
2. **Another SHACS speaker from IBM Federal Systems Division who graduated over 20 years ago and has been the Chief Software Engineer at Motorola – Arizona: The most important thing I learned at SHSU was table driven problem solvers.**
3. **CS major alumni now working as a federal prosecutor in the Federal Post Office: The most important thing I learned at SHSU was problem decomposition (especially decision tables, finite state automata, and table driven methods).**

I credit the SHSU Computer Science program for making me exercise my mind to conquer complex problems—problem decomposition. Those analytical skills have enabled me to take complex labor cases, disaggregate the essential facts so I can assimilate a cogent, indefeasible argument. (I get to ‘play’ law without a law degree.) Seriously, I write: position papers, present opening arguments, conduct direct examination and cross examination of witnesses, make closing arguments, and write closing briefs. I work closely with the staff attorneys who are down the hall. However, when I go to hearing, **I represent the USPS by myself—no attorney (exception noted for the hearing officer) present. I have a 95% win rate on discipline cases, 90% win rate on contract cases, 100% win rate on Employee Debt Collections, and 85% win rate on Unemployment Benefit Hearings**—very hard to win when the state statutes create an entitlement. These are by far the best success rates in the department.

1. **The space shuttle contained approximately 1.5 million lines of code. About 150, 000 lines must be tailored for each mission. New task can frequently be built from portions of previous task. A single mistake could result in the loss of a vehicle and its crew. Errors simply cannot be tolerated. The coding problem is solved by not coding but rather using table drive problem solvers.**

**COMMUNICATIONS**

**Sign from New Branufels Wuerstfest.**

**I hope you understand that what I really said was not what I meant but what I really thought was what I really wanted to say.**

**&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&**

**Customers find it very difficult to organize and express their desires and recognize real needs!**

**To be successful as a software professional you frequently need to become a subject expert in the customer’s field!**

**What is the difference in the following?**

**1) Add A to B unless A is less than B, in which case subtract A from B.**

**2) Add A to B. However, if A is less than B, the answer is the difference of A and B.**

**3) Add A to B, but subtract A from B where A is less than B.**

**4) The total is found by adding B to A. Not withstanding the previous sentence, in the circumstance where B is greater than A the result is the difference between B and A.**

**5) The total is the sum of A and B: only when A is less than B should the difference be used as the total.**

**6) IF A < B THEN**

**SUBTRACT A FROM B**

**ELSE**

**ADD A TO B.**

***Statement of Business Policy***

**Customer who order more than $10,000 in products per year and in addition have a good payment history are to receive priority treatment; if their payment history is poor, they should only receive priority treatment if they have been with us for more than 20 years. Other customers are to receive normal treatment unless they have a good payment history in which case they are entitled to express treatment.**



**Decision Tree**

**Omissions, Contradictions, Redundancies, Filter Out Inappropriate Information?**

**Types of Intelligence**

1. **Graphical – a picture is worth a thousand words (artistic/managerial).**
2. **Symbolic Manipualtive –algebraic/abstract (mathematical/engineering).**

**(Left Brain versus Right Brain -**

**Artistic versus Scientific/Engineering)**

**The most successful entrapuneurs learn to combine both types of intelligence!**

**IF (question)**

**TRUE**

**ELSE**

**FALSE;**

**END IF;**

**ALTERNATIVES**

**1) Pseudo Code**

**2) Decision Trees**

**3) Flow Charts**

**4) PDL (program design language)**

1. **Formal Design Methodologies**
2. **Table Driven Problem Solvers (especially Decision Tables and Finite State Automata)**

**Basic Decision Table Elements**

**Condition Condition Entries**

**Stubs**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | **Rule 1** | **Rule 2** | **Rule 3** | **Rule 4** |
| **IF** |  |  |  |  |  |
| **and** |  |  |  |  |  |
| **and** |  |  |  |  |  |
| **and** |  |  |  |  |  |
| **THEN** |  |  |  |  |  |
| **action** |  |  |  |  |  |
| **action** |  |  |  |  |  |

**Action Action Entries**

**Stubs**

**Customer who order more than $10,000 in products per year and in addition have a good payment history are to receive priority treatment; if their payment history is poor, they should only receive priority treatment if they have been with us for more than 20 years. Other customers are to receive normal treatment unless they have a good payment history in which case they are entitled to express treatment.**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **more than $10,000** |  | **y** | **y** | **y** | **y** | **n** | **n** | **n** | **n** |
| **good pay history** |  | **y** | **y** | **n** | **n** | **y** | **y** | **n** | **n** |
| **more than 20 years** |  | **y** | **n** | **y** | **n** | **y** | **n** | **y** | **n** |
|  |  |  |  |  |  |  |  |  |  |
| **priority treatment** |  | **x** | **x** | **x** |  |  |  |  |  |
| **express treatment** |  |  |  |  |  | **x** | **x** |  |  |
| **normal treatment** |  |  |  |  | **x** |  |  | **x** | **x** |

***Limited Entry Decision Table***

**The number of entries is 2\*\*N, where N is the number of conditions.**

**1 2 3 4**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Credit limit is satisfactory** |  | **y** | **n** | **n** | **n** |
| **Pay experience is favorable** |  | **-** | **y** | **n** | **n** |
| **Special clearance obtained** |  | **-** | **-** | **y** | **n** |
|  |  |  |  |  |  |
| **Approve Order** |  | **x** | **x** | **x** |  |
| **Reject Order** |  |  |  |  | **x** |

**Limited Entry Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Credit limit** |  | **yes** | **no** | **maybe** |
| **Pay experience** |  | **no** | **yes** | **no** |
| **Special clearance** |  | **no** | **ok** | **yes** |
|  |  |  |  |  |
| **approve** |  |  |  | **x** |
| **reject** |  | **x** | **x** |  |

**Extended/Mixed Entry Table**

**Example, spousal responces are not necessarily binary for binary questions. They tend to be ternary – yes, no and maybe!**

**R1 R2 R3 R4 R5**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Condition 1** |  | **Y** | **Y** | **Y** | **Y** | **Y** |
| **Condition 2** |  | **Y** | **N** | **N** | **N** | **N** |
| **Condition 3** |  | **N** | **N** | **Y** | **N** | **Y** |
|  |  |  |  |  |  |  |
| **Action 1** |  | **x** |  | **x** |  | **x** |
| **Action 2** |  |  | **x** |  | **x** | **x** |

**Redundancy: R2 and R4**

**Contradiction: R3 and R5**

**=======================================**

**Completeness for (binary) Limited Entry Tables:**

**The number of columns = 2\*\*N where N is the number of conditions.**

**Pollack, S. L., Hicks, H., and Harrison, W. J. "Decision Tables: Theory and Practice," Wily, New York, 1971.**

**Mixed Entry Tables:**

**(2 responces) 3 questions = 8 possibilities**

**(3 responces) 2 questions = 9 possibilities**

**(3 responces) 3 questions = 27 possibilities**

**Assume a test with five questions. Three questions have binary responces and two questions have ternary responces. Then:**

**(2 responces) 3 questions \* (3 responces) 2 questions = 8 \* 9 = *72 possibilities must be addressed*!**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| **Credit limit ok** |  | **y** | **y** | **y** | **y** | **n** | **n** | **n** | **n** |
| **Pay experience ok** |  | **y** | **y** | **n** | **n** | **y** | **n** | **y** | **n** |
| **Special clearance** |  | **y** | **n** | **y** | **n** | **y** | **y** | **n** | **n** |
|  |  |  |  |  |  |  |  |  |  |
| **Approve Order** |  | **x** | **x** | **x** | **x** | **x** | **x** | **x** |  |
| **Reject Order** |  |  |  |  |  |  |  |  | **x** |

***Full Table - Specification***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Credit limit ok** |  | **y** | **-** | **-** | **n** |
| **Pay experience ok** |  | **-** | **y** | **-** | **n** |
| **Special clearance** |  | **-** | **-** | **y** | **n** |
|  |  |  |  |  |  |
| **Approve Order** |  | **x** | **x** | **x** |  |
| **Reject Order** |  |  |  |  | **x** |

**Reduced Table Due to Redundancy of Actions - Design**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **condition 1** |  | **y** | **y** | **n** | **-** |
| **condition 2** |  | **y** | **n** | **y** | **-** |
| **condition 3** |  | **n** | **y** | **y** | **-** |
| **condition 4** |  | **y** | **n** | **y** | **-** |
|  |  |  |  |  |  |
| **action 1** |  | **x** |  |  | **x** |
| **action 2** |  |  | **x** | **x** | **x** |
| **action 3** |  | **x** |  | **x** | **x** |

**Note rules = 2\*\*4 = 16 but only three (3) posibilities are meaningful.**

**The last column is a catch-all, an "ELSE."**

**Bartender: What would you like? Customer: I don’t know, what can you offer?**

**1 2 3 4 5 6 7 8**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **bourbon base** |  | **y** | **y** | **y** | **n** | **y** | **n** | **n** | **n** |
| **gin base** |  | **y** | **y** | **n** | **y** | **n** | **y** | **n** | **n** |
| **vodka base** |  | **y** | **n** | **y** | **y** | **n** | **n** | **y** | **n** |
|  |  |  |  |  |  |  |  |  |  |
| **offer vodka or gin & vermouth (Martini)** |  | **x** | **x** | **x** | **x** |  | **x** | **x** |  |
| **offer vodka & orange juice (screwdriver)** |  | **x** |  | **x** | **x** |  |  | **x** |  |
| **offer vodka & tomato juice (Bloody Mary)** |  | **x** |  | **x** | **x** |  |  | **x** |  |
| **offer bourbon and vermouth (Manhattan)** |  | **x** | **x** | **x** |  | **x** |  |  |  |
| **offer bourbon & water (Old Fashion)** |  | **x** | **x** | **x** |  | **x** |  |  |  |
| **offer a soft drink/bottle water** |  |  |  |  |  |  |  |  | **x** |

**1 2 3 4 5 6**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **bourbon** |  | **y** | **y** | **n** | **y** | **n** | **n** |
| **gin** |  | **I** | **y** | **I** | **n** | **y** | **n** |
| **vodka** |  | **y** | **n** | **y** | **n** | **n** | **n** |
|  |  |  |  |  |  |  |  |
| **offer vodka or gin & vermouth (Martini)** |  | **x** | **x** | **x** |  | **x** |  |
| **offer vodka & orange juice (screwdriver)** |  | **x** |  | **x** |  |  |  |
| **offer vodka & tomato juice (Bloody Mary)** |  | **x** |  | **x** |  |  |  |
| **offer bourbon and vermouth (Manhattan)** |  | **x** | **x** |  | **x** |  |  |
| **offer bourbon & water (Old Fashion)** |  | **x** | **x** |  | **x** |  |  |
| **offer a soft drink/bottled water** |  |  |  |  |  |  | **x** |

**Note rows 1 and 3 collapsed to new row 1.**

**Note rows 4 and 7 collapsed to new row 3.**

**CHECK FOR COMPLETENESS**

**1) Be sure each rule has at least one action. 2) Make sure each pair of rules is independent (all have at least one N for another functions Y). 3) Show that the smaller table will expand to the same size -- recall that for each "AND" an "I" in r positions is equal 2\*\*r rules.**

**R1 = 2\*\*1 = 2, R2 = 2\*\*0 = 1, R3 = 2\*\*1 = 2, R4 = 2\*\*0 = 1, R5 = 2\*\*0 = 1, R6 = 2\*\*0 = 1.**

**Hence R1+R2+R3+R4+R5+R6 = 2+1+2+1+1+1=8.**

**POLICY STATEMENT:**

**We have decided to implement a new policy for granting credit based on the following. Please implement this ASAP (PDQ) or collect your severance pay. Basically we plan to limit credit to an individuals discretionary income (whats left after all expenses) multiplied by an appropriate multiplier. The multiplier is to be determined as follows:**

**For married home owners who have retained their present job for two or more years, 5 times.**

**For married couples who are either home owners or have maintained their current job for two or more years, 4 times.**

**For unmarried home owners who have retained their current job for two or more years, 3 times.**

**For married persons who rent and for unmarried home owners who have been employeed at their current job less than two years, 2 times.**

**For single, widowed, or divorced persons who rent, 1 time.**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **MARRIED** |  | **Y** | **Y** | **Y** | **Y** | **N** | **N** | **N** | **N** |
| **OWNED** |  | **Y** | **Y** | **N** | **N** | **Y** | **Y** | **N** | **N** |
| **>= 2 YRS** |  | **Y** | **W N** | **Y** | **N** | **Y** | **N** | **Y** | **N** |
|  |  |  |  |  |  |  |  |  |  |
| **CREDIT 1** |  |  |  |  |  |  |  | **X** | **X** |
| **CREDIT 2** |  |  |  |  | **X** |  | **X** |  |  |
| **CREDIT 3** |  |  |  |  |  | **X** |  |  |  |
| **CREDIT 4** |  |  | **X** | **X** |  |  |  |  |  |
| **CREDIT 5** |  | **X** |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **MARRIED** |  | **Y** | **Y** | **Y** | **Y** | **N** | **N** | **N** | **N** |
| **OWNED** |  | **Y** | **Y** | **N** | **N** | **Y** | **Y** | **N** | **N** |
| **>= 2 YRS** |  | **Y** | **N** | **Y** | **N** | **Y** | **N** | **Y** | **N** |
|  |  |  |  |  |  |  |  |  |  |
| **CREDIT 1** |  |  |  |  |  |  |  | **X** | **X** |
| **CREDIT 2** |  |  |  |  | **X** |  | **X** |  |  |
| **CREDIT 3** |  |  |  |  |  | **X** |  |  |  |
| **CREDIT 4** |  |  | **X** | **X** |  |  |  |  |  |
| **CREDIT 5** |  | **X** |  |  |  |  |  |  |  |

**IF MARRIED**

**IF OWNED**

**IF YEARS >= 2**

**MOVE 5 TO CREDIT-FACTOR**

**ELSE**

**MOVE 4 TO CREDIT-FACTOR**

**ELSE**

**IF YEARS >= 2**

**MOVE 4 TO CREDIT-FACTOR**

**ELSE**

**MOVE 2 TO CREDIT-FACTOR**

**ELSE**

**IF OWNED**

**IF YEARS-EMPLOYED >= 2**

**MOVE 3 TO CREDIT-FACTOR**

**ELSE**

**MOVE 2 TO CREDIT-FACTOR**

**ELSE**

**IF YEARS-EMPLOYED >= 2**

**MOVE 1 TO CREDIT-FACTOR**

**ELSE**

**MOVE 1 TO CREDIT-FACTOR.**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **MARRIED** |  | **Y** | **Y** | **Y** | **Y** | **N** | **N** | **N** | **N** |
| **OWNED** |  | **Y** | **Y** | **N** | **N** | **Y** | **Y** | **N** | **N** |
| **>= 2 YRS** |  | **Y** | **N** | **Y** | **N** | **Y** | **N** | **Y** | **N** |
|  |  |  |  |  |  |  |  |  |  |
| **CREDIT 1** |  |  |  |  |  |  |  | **X** | **X** |
| **CREDIT 2** |  |  |  |  | **X** |  | **X** |  |  |
| **CREDIT 3** |  |  |  |  |  | **X** |  |  |  |
| **CREDIT 4** |  |  | **X** | **X** |  |  |  |  |  |
| **CREDIT 5** |  | **X** |  |  |  |  |  |  |  |

**COBOL has world’s most powerful “CASE” structure!**

**EVALUATE MARRIED ALSO OWNED ALSO YR >=2**

**WHEN "Y" ALSO "Y" ALSO "Y" PERFORM CD5**

**WHEN "Y" ALSO "Y" ALSO "N" PERFORM CD4**

**WHEN "Y" ALSO "N" ALSO "Y" PERFORM CD4**

**WHEN "Y" ALSO "N" ALSO "N" PERFORM CD2**

**WHEN "N" ALSO "Y" ALSO "Y" PERFORM CD3**

**WHEN "N" ALSO "Y" ALSO "N" PERFORM CD2**

**WHEN "N" ALSO "N" ALSO "Y" PERFORM CD1**

**WHEN "N" ALSO "N" ALSO "N" PERFORM CD1**

**END EVALUATE.**

**EVALUATE OFFICE**

**WHEN "A01" MOVE 1 TO REGION**

**WHEN "A02" THRU "C16" MOVE 2 TO REGION**

**WHEN OTHER MOVE 0 TO REGION**

**END EVALUATE.**

**EVALUATE LOW-STOCK ALSO WEEKLY-USE ALSO LOCAL-VENDOR ALSO ON-ORDER**

**WHEN "Y" ALSO 16 THRU 999 ALSO ANY ALSO "N" GO TO RUSH-ORDER**

**WHEN "Y" ALSO 16 THRU 768 ALSO ANY ALSO "Y" GO TO NORMAL**

**WHEN "Y" ALSO 8 THRU 15 ALSO "N" ALSO "N" GO TO RUSH-ORDER**

**WHEN "N" ALSO ANY ALSO ANY ALSO "Y" GO TO CANCEL**

**WHEN ANY ALSO ANY ALSO ANY ALSO ANY GO TO REJECT**

**END EVALUATE.**

**IDENTIFICATION DIVISION. \*> in file DecTab2.CBL**

**PROGRAM-ID. DecisionTable.**

**AUTHOR. Burris.**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

***\* Policy statement: We plan to limit credit to an individuals***

***\* discretionary income (whats left after expenses) multiplied***

***\* by an appropriate multiplier. The multiplier is to be***

***\* determined as follows:***

***\* 1) For married home owners who have retaine their current job***

***\* for two or more years, five.***

***\* 2) For married couples who either own their own home or have***

***\* maintained their current jor for two or more years, four.***

***\* 3) For unmarried home owners who have retained their current***

***\* job for two or more years, three.***

***\* 4) For married couples who rent and for unmarried home owners***

***\* who have retained their current job less than two years, one.***

***\* 5) For single, widowed, or divorced persons who rent, one.***

***\****

***\** Decision Table**

***\* MARRIED ||Y|Y|Y|Y|N|N|N|N***

***\* OWN ||Y|Y|N|N|Y|Y|N|N***

***\* >= 2 YR ||Y|N|Y|N|Y|N|Y|N***

***\* ==========================***

***\* CREDIT 1|| | | | | | |X|X***

***\* CREDIT 2|| | | |X| |X| |***

***\* CREDIT 3|| | | | |X| | |***

***\* CREDIT 4|| |X|X| | | | |***

***\* CREDIT 5||X| | | | | | |***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**ENVIRONMENT DIVISION.**

**INPUT-OUTPUT SECTION.**

**FILE-CONTROL.**

**DATA DIVISION.**

**FILE SECTION.**

**WORKING-STORAGE SECTION.**

**01 MoreTransactions PIC XXX.**

**01 DecisionTableVariables.**

**10 Married Pic X.**

**10 Own Pic X.**

**10 Years Pic 99.**

**10 Finished Pic XXX VALUE "No".**

**01 OfficeVariables.**

**10 Office PIC XXX VALUE "B17".**

**10 Region PIC 9.**

**01 InventoryVariables.**

**10 LowStock PIC X VALUE "Y".**

**10 WeeklyUse PIC 999 VALUE 456.**

**10 LocalVendor PIC X VALUE "N".**

**10 OnOrder PIC X VALUE "N".**

**PROCEDURE DIVISION.**

**DEMONSTRATE-DECISION-TABLE SECTION.**

**MOVE "Yes" TO MoreTransactions.**

**PERFORM UNTIL MoreTransactions = "No" or "no"**

**DISPLAY "Process another transaction? (Yes or No): "**

**ACCEPT MoreTransactions**

**EVALUATE MoreTransactions**

**WHEN "Yes" PERFORM OBTAIN-REQUEST**

**PERFORM PROCESS-Request**

**WHEN "yes" PERFORM OBTAIN-REQUEST**

**PERFORM PROCESS-Request**

**WHEN "No" DISPLAY "Have a nice day :-)."**

**WHEN "no" DISPLAY "Have a nice day :-)."**

**WHEN ANY DISPLAY "Illegal responce, try again."**

**END-EVALUATE**

**END-PERFORM.**

**PERFORM EVALUATE-SAMPLES.**

**STOP RUN.**

**OBTAIN-REQUEST SECTION.**

**GetResponces.**

**DISPLAY "Married ('Y' for yes, 'N' for no)? ".**

**ACCEPT Married.**

**If Married = 'y' MOVE "Y" TO Married.**

**If Married = "n" MOVE "N" TO MARRIED.**

**DISPLAY "Own dwelling ('Y' for yes, 'N' for no)? ".**

**ACCEPT Own.**

**If Own = 'y' MOVE "Y" TO Own.**

**If Own = "n" MOVE "N" TO Own.**

**DISPLAY "Enter (integer) years at current job. ".**

**ACCEPT Years.**

**PROCESS-REQUEST SECTION.**

**EVALUATE-DECISION-TABLE. \*> Hard code solution.**

**EVALUATE Married ALSO Own ALSO Years >= 2**

**WHEN "Y" ALSO "Y" ALSO TRUE PERFORM CREDIT5**

**WHEN "Y" ALSO "Y" ALSO FALSE PERFORM CREDIT4**

**WHEN "Y" ALSO "N" ALSO TRUE PERFORM CREDIT4**

**WHEN "Y" ALSO "N" ALSO FALSE PERFORM CREDIT2**

**WHEN "N" ALSO "Y" ALSO TRUE PERFORM CREDIT3**

**WHEN "N" ALSO "Y" ALSO FALSE PERFORM CREDIT2**

**WHEN "N" ALSO "N" ALSO TRUE PERFORM CREDIT1**

**WHEN "N" ALSO "N" ALSO FALSE PERFORM CREDIT1**

**END-EVALUATE.**

**GO TO EXIT-EVALUATE-DECISION-TABLE.**

**CREDIT1.**

**DISPLAY "CREDIT FACTOR 1 APPROVED.".**

**CREDIT2.**

**DISPLAY "CREDIT FACTOR 2 APPROVED.".**

**CREDIT3.**

**DISPLAY "CREDIT FACTOR 3 APPROVED.".**

**CREDIT4.**

**DISPLAY "CREDIT FACTOR 4 APPROVED.".**

**CREDIT5.**

**DISPLAY "CREDIT FACTOR 5 APPROVED.".**

**EXIT-EVALUATE-DECISION-TABLE.**

**EXIT.**

**\* Process another transaction? (Yes or No):**

**\* yes**

**\* Married (‘Y’ for yes, “N’ for no)?**

**\* y**

**\* Own dwelling (‘Y’ for yes, “N’ for no)?**

**\* y**

**\* Enter (integer) years at current job.**

**\* 6**

**\* CREDIT FACTOR 5 APPROVED.**

**\* Process another transaction? (Yes or No):**

**\* no**

**\* Have a nice day :-).**

**EVALUATE-SAMPLES SECTION.**

**EVALUATE Office**

**WHEN "A01" MOVE 1 TO Region**

**WHEN "A02" THRU "C16" MOVE 2 TO Region**

**WHEN OTHER MOVE 0 TO Region**

**END-EVALUATE.**

**DISPLAY "Region = " Region.**

**EVALUATE LowStock ALSO WeeklyUse ALSO LocalVendor ALSO OnOrder**

**WHEN "Y" ALSO 16 THRU 999 ALSO ANY ALSO "N" GO TO RushOrder**

**WHEN "Y" ALSO 13 THRU 487 ALSO ANY ALSO "Y" GO TO NormalOrder**

**WHEN "Y" ALSO 8 THRU 15 ALSO "N" ALSO "N" GO TO RushOrder**

**WHEN "N" ALSO ANY ALSO ANY ALSO "Y" GO TO CancelOrder**

**WHEN ANY ALSO ANY ALSO ANY ALSO ANY GO TO RejectOrder**

**END-EVALUATE.**

**RushOrder.**

**DISPLAY "Rush order.". GO TO EVALUATE-SAMPLE-EXIT.**

**NormalOrder.**

**DISPLAY "Normal order.". GO TO EVALUATE-SAMPLE-EXIT.**

**CancelOrder.**

**DISPLAY "Cancel order.". GO TO EVALUATE-SAMPLE-EXIT.**

**RejectOrder.**

**DISPLAY "Reject order.".**

**EVALUATE-SAMPLE-EXIT.**

**EXIT.**

**\* Region = 2**

**\* Rush order.**

***HARD CODE PROBLEM:***

**Every time rule values or required action sequences change the code must be modified and the new code validated.**

**Examples: The space shuttle has approximately 1.5 million lines of code. About 150,000 must be tailored to each mission. How does one accomplish this massive change reliabily? How to code the Air Force Reserve Payroll: world wide with small variations at each location.**

***SOLUTION:***

**Store the decision process as a table.**

**Accept user responces and store as a match vector.**

**Compare the match vector against the process table until a match is found then perform the corresponding actions.**

**To implement new strategies, change the table, not the code.**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **C1** |  | **Y** | **N** | **Y** |  | **C1** |  | **Y** | **N** | **N** |
| **C2** |  | **N** | **Y** | **Y** |  | **C2** |  | **N** | **Y** | **N** |
| **C3** |  | **Y** | **Y** | **Y** |  | **C3** |  | **Y** | **Y** | **N** |
|  |  |  |  |  |  |  |  |  |  |  |
| **A1** |  | **X** |  | **X** |  | **A1** |  |  |  | **X** |
| **A2** |  |  | **X** | **X** |  | **A2** |  | **X** | **X** |  |

**STRATEGY 1 STRATEGY 2**

**"Table Drive Proble Solvers" by David Burris and Kurt Schember, 18th Annual Southwest Regional ACM Conference Proceedings, 1980.**

***CREDIT POLICY:***

**If the applicant has special clearance, approve the loan (S4). If the applicant does not have special clearance and their credit is not okay, reject the loan (S5). In the case their credit is okay then check their pay experience (do they make payments on time). If the pay experience is good, approve the loan (S3). If the pay experience is fair, approve the loan but notify the credit manager (S7). If they are slow to pay (poor) reject the loan (S6).**



***Deterministic Finite State Automata***

|  |
| --- |
|  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Y** | **N** | **FAIR** |
| **Special Clearance** | **0** | **4** | **1** | **8** |
| **Credit Ok** | **1** | **2** | **5** | **8** |
| **Pay Experience** | **2** | **3** | **6** | **7** |

***State Table***

**procedure decide(p: in answer) is**

**next\_state: integer := 0; finished: boolean := false;**

**state\_tab: array(0..2, responce\_type) of integer := ( (4, 1, 8), (2, 5, 8), (3, 6, 7) );**

**begin**

**while not finished loop**

**case next\_state is**

**when 0 => next\_state := state\_tab(0, p(1));**

**when 1 => next\_state := state\_tab(1, p(2));**

**when 2 => next\_state := state\_tab(2, p(3));**

**when 3 => put("Approve the order."); new\_line;**

**finished := true;**

**when 4 => put("Approve the order and notify parent!");**

**new\_line; finished := true;**

**when 5 => put("Reject the order and kick the bum out!");**

**new\_line; finished := true;**

**when 6 => put("Reject the order!!"); new\_line;**

**finished := true;**

**when 7 => put("Approve the order and notify management.");**

**new\_line; finished := true;**

**when 8 => put("This is an error routine!"); finished := true;**

**when others => put("Illegal integer input."); finished := true;**

**end case;**

**end loop;**

**end decide;**

**begin**

**responce(1) := no; responce(2) := yes; responce(3) := fair;**

**decide(responce);**

**end;**

**Approve the order and notify management!**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Y** | **N** | **FAIR** |
| **Special Clearance** | **0** | **4** | **1** | **8** |
| **Credit Ok** | **1** | **2** | **5** | **8** |
| **Pay Experience** | **2** | **3** | **6** | **7** |

***State Table***

with ada\_io; use ada\_io;

**procedure table\_driven is**

**type responce\_type is (yes, no, fair);**

**type answer is array(1..3) of responce\_type;**

**responce: answer;**

**procedure decide(p: in answer) is**

**next\_state: integer := 0;**

**finished: boolean := false;**

**state\_tab: array(0..2, responce\_type) of integer := ( (4, 1, 8), (2, 5, 8), (3, 6, 7) );**

**begin**

**while not finished loop**

**case next\_state is**

**when 0 => next\_state := state\_tab(0, p(1));**

**when 1 => next\_state := state\_tab(1, p(2));**

**when 2 => next\_state := state\_tab(2, p(3));**

**when 3 => put("Approve the order."); new\_line; finished := true;**

**when 4 => put("Approve the order and notify parent!");**

**new\_line; finished := true;**

**when 5 => put("Reject the order and kick the bum out!");**

**new\_line; finished := true;**

**when 6 => put("Reject the order!!"); new\_line; finished := true;**

**when 7 => put("Approve the order and notify management.");**

**new\_line; finished := true;**

**when 8 => put("This is an error routine!"); finished := true;**

**when others => put("Illegal integer input."); finished := true;**

**end case;**

**end loop;**

**end decide;**

**begin**

**responce(1) := yes; responce(2) := no; responce(3) := no;**

**decide(responce);**

**responce(1) := no; responce(2) := no; responce(3) := no;**

**decide(responce);**

**responce(1) := no; responce(2) := yes; responce(3) := yes;**

**decide(responce);**

**responce(1) := no; responce(2) := yes; responce(3) := fair;**

**decide(responce);**

**end;**

**Approve the order and notify and notify parent!**

**Reject the order and kick the bum out!**

**Approve the order.**

**Approve the order and notify management.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Y** | **N** | **FAIR** |
| **Special Clearance** | **0** | **4** | **1** | **8** |
| **Credit Ok** | **1** | **2** | **5** | **8** |
| **Pay Experience** | **2** | **3** | **6** | **7** |



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Y** | **N** | **FAIR** |
| **Special Clearance** | **0** | **4** | **1** | **8** |
| **Credit Ok** | **1** | **2** | **5** | **8** |
| **Pay Experience** | **2** | **3** | **6** | **7** |

***State Table***



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Y** | **N** | **FAIR** |
| **Special Clearance** | **0** | **4** | **1** | **8** |
| **Credit Ok** | **1** | **2** | **5** | **8** |
| **Pay Experience** | **2** | **3** | **6** | **5** |

***State Table***