The Tax Forms Advisor

Develop an expert system which recommends the appropriate income tax forms, after asking the user a number of questions.

Collect requirements using:
- Interviews
- Documents and other sources.

Forms

- **Form 1040**: The standard form.
- **Form 1040A**: The short form. Use instead of 1040 if taxable income is less than $50,000. Deductions cannot be itemised using this form, but you can get a credit for child-care expenses.
- **Form 1040EZ**: Use instead of 1040A if income less than $50,000, you have no dependents, and deductions are not itemised. If you are married, your spouse must file a join return or you can’t use this form. This form cannot be used if you earned more than $400 in taxable interest.
- **Form 2441**: Lets you claim a credit for daycare expenses. Also, credit for care of an elderly parent or other dependent.
- **Form 2016EZ**: Deduct the unreimbursed part of any expenses your incurred for your employer (primarily travel, excluding commuting).
- **Form 3903**: Deduction for unreimbursed moving expenses if you moved this year because of your job. Also, if you changed work locations and the new workplace is more than 50 miles further from your old home than your old workplace.
- **Form 4684**: Recover some of the losses during the year, the part not covered by insurance. Deduct costs due to car accidents.
- **Form 4868**: Get an extension for filing your taxes.
- **Form 8283**: Get credit for donating more than $500 worth of property to charity.
- **Form 8829**: Deduct expenses associates with a home office.
Design
deftemplate types:
  • form
  • user
  • question
  • answer
  • recommendation

Asking the User
(deffunction ask-user (?question ?type)
  "Ask a question, and return the answer"
  (bind ?answer "")
  (while (not (is-of-type ?answer ?type)) do
    (printout t ?question " ")
    (if (eq ?type yes-no) then
      (printout t "(yes or no) ")
    (bind ?answer (read)))
  (return ?answer))

(deffunction is-of-type (?answer ?type)
  "Check that the answer has the right form"
  (if (eq ?type yes-no) then
    (return (or (eq ?answer yes) (eq ?answer no)))
  else (if (eq ?type number) then
    (return (numberp ?answer)))
  else (return (> (str-length ?answer) 0))))

(defrule ask::ask-question-by-id
  "Given the identifier of a question, ask it and assert the answer"
  (declare (auto-focus TRUE))
  (MAIN::question (ident ?id) (text ?text) (type ?type))
  (not (MAIN::answer (ident ?id)))
?ask <- (MAIN::ask ?id)
=>
  (bind ?answer (ask-user ?text ?type))
  (assert (answer (ident ?id) (text ?answer)))
  (retract ?ask)
  (return))

(deftemplate user
  (slot income (default 0))
  (slot dependents (default 0)))

(deftemplate question
  (slot text)
  (slot type)
  (slot ident))

(deftemplate answer
  (slot ident)
  (slot text))

(deftemplate recommendation
  (slot form)
  (slot explanation))
Preparing the Questions

(deffacts question-data
 "The questions the system can ask."
 (question (ident income) (type number)
   (text "What was your annual income?")
 (question (ident interest) (type yes-no)
   (text "Did you earn more than $400 of taxable interest?"
 (question (ident dependents) (type number)
   (text "How many dependents live with you?")
 (question (ident childcare) (type yes-no)
   (text "Did you have dependent care expenses?")
 (question (ident moving) (type yes-no)
   (text "Did you move for job-related reasons?")
 (question (ident employee) (type yes-no)
   (text "Did you have unreimbursed employee expenses?")
 (question (ident reimbursed) (type yes-no)
   (text "Did you have reimbursed employee expenses, too?")
 (question (ident casualty) (type yes-no)
   (text "Did you have losses from a theft or an accident?"
 (question (ident on-time) (type yes-no)
   (text "Will you be able to file on time?"
 (question (ident charity) (type yes-no)
   (text "Did you give more than $500 in property to charity?"
 (question (ident home-office) (type yes-no)
   (text "Did you work in a home office?")

Forcing the Questions to be Asked

(defrule request-income
 => (assert (ask income)))
(defrule request-num-dependents
 => (assert (ask dependents)))
(defrule assert-user-fact
 (answer (ident income) (text ?i))
 (answer (ident dependents) (text ?d))
 =>
 (assert (user (income ?i) (dependents ?d))))
(defrule request-interest-income
 ;; If the total income is less than 50000
 (answer (ident income) (text ?i&:(< ?i 50000)))
 =>
 (assert (MAIN::ask interest)))
(defrule request-childcare-expenses
 ;; If the user has dependents
 (answer (ident dependents) (text ?d&:(> ?d 0)))
 =>
 (assert (ask childcare)))
(defrule request-employee-expenses
 =>
 (assert (ask employee)))
(defrule request-reimbursed-expenses
 ;; If there were unreimbursed employee expenses...
 (answer (ident employee) (text ?t&:(eq ?t yes)))
 =>
 (assert (ask reimbursed))

Recommendations

(defrule combine-recommendations
  ?r1 <- (recommendation (form ?f) (explanation ?e1))
  ?r2 <- (recommendation (form ?f) (explanation ?e2&:(neq ?e1 ?e2)))
=> (retract ?r2)
  (modify ?r1 (explanation (str-cat ?e1 *crlf* ?e2))))

(defrule form-1040EZ
  (user (income ?i&:(< ?i 50000))
       (dependents ?d&:(eq ?d 0)))
  (answer (ident interest) (text no))
=> (assert (recommendation (form 1040EZ)
                          (explanation "Income below threshold, no dependents"))))

(defrule form-1040A
  (user (income ?i&:(< ?i 50000))
       (dependents ?d&:(> ?d 0)))
=> (assert (recommendation (form 1040A)
                          (explanation "Income below threshold, with dependents"))))

(defrule form-1040A-excess-interest
  (user (income ?i&:(< ?i 50000))
       (answer (ident interest) (text yes)))
=> (assert (recommendation (form 1040A)
                          (explanation "Excess interest income"))))

(defrule form-1040-income-above-threshold
  (user (income ?i&:(>= ?i 50000)))
=> (assert (recommendation (form 1040A)
                          (explanation "Income above threshold"))))
(assert (recommendation
    (form 1040)
    (explanation "Income above threshold")))

...