This is a brief document which explains certain aspects of the implementation document “paramstyle12.ml”.

Used a list data structure to hold the various passing styles and used a record to represent specific passing styles which consist of passing style name and factors. Also, used a record to hold the various factors of the passing styles and used a list to hold instance/instances for factors that effects parameter passing.

Determining or defining a passing style depends on the values of these factors that are assumed to effect parameter passing style. When the code is first run, users are allowed to select (give) the factors type they are aware of by giving a number that correspond to the factor as indicated by a help menu that is first displayed. After selecting the factor, users are allowed (asked) to initialize the selected factor. After initializing the selected factor, users are asked if they wish to add more factors for the passing style. If these selected factors and their values corresponds to factors and values of known passing styles then the passing style is return alongside it properties (name and factors) and the effects induced by these values. If values do not corresponds to values for known passing style then a new passing style is created with an input name for the passing style and is added to the list of passing styles. This is illustrated below.

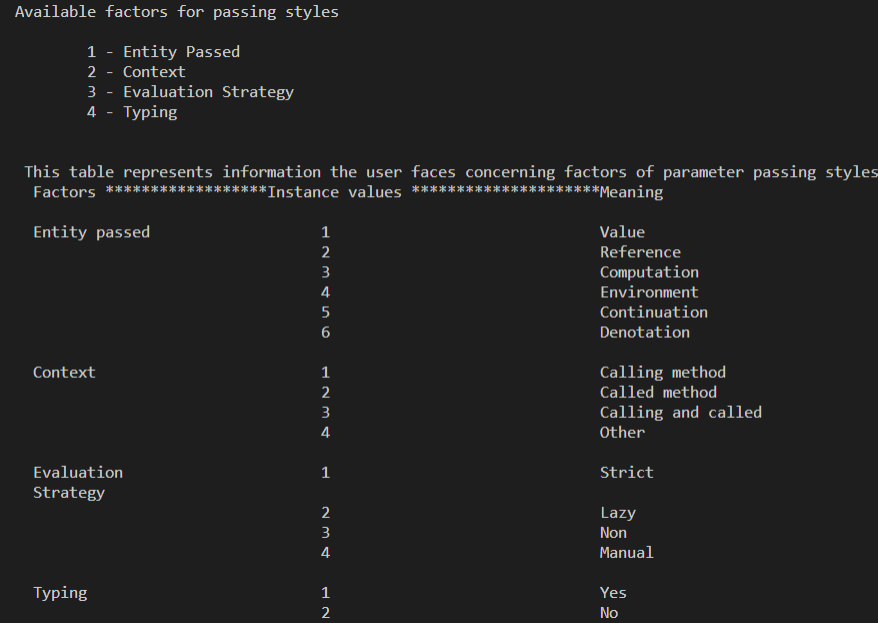


Fig.1. Factor numbers and meaning of instances keyed in.

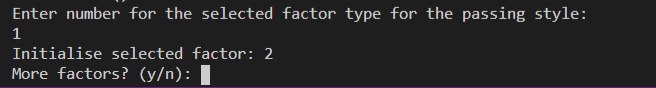


Fig.2. Selecting factor and giving an instance of the selected factor

If users goes for more factors (entering yes or y) then the users are allow to give another factor for the passing style and initializing the selected factor. This can be seen below.

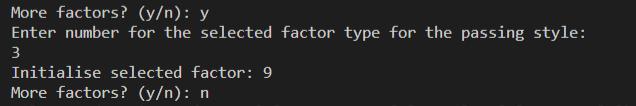


Fig.3.

Also a specific factor can be given more than one instance by selecting the factor more than one time and giving an instance (value) for all the selected case of the factor. The initialized instances for the factor are held in the instance list for that factor (e.g. as in ent below). For factors not of interest to the user, a default value would be assigned to this factor (the number 5 in this case). This can be seen as indicated below.

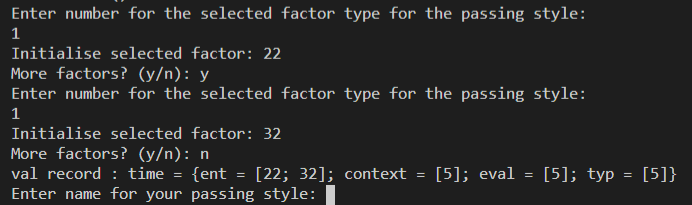


Fig.4. Selecting factor 1 (entity\_type) more than one time.

When users go for no more factors, if selected factor(s) and initialized value(s) correspond to that of known passing style (e.g. 1, 1, 1, 1 for all four factors), then the passing style name is displayed alongside it properties (name and factors with their values) after which it has displayed the list of passing style.

For values of selected factor(s) that does not correspond to that of known passing style (e.g. 8, 8, for any two factors), a name is demanded for the passing style and a new passing style is constructed with those values for the selected factors and add the passing style to the list of passing styles.

After adding/displaying the passing style to/from list of styles, a select menu is then displayed asking the user if they want more passing style. If users go for yes or y then a submenu is displayed giving them options (operations) from which they can select from and it’s immediately initiated once it is selected. If the users go for no, the final list of passing styles is displayed which contain the new passing style added if users defines their own new style. This can be seen as indicated below.

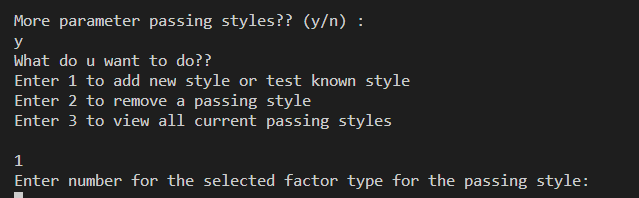


Fig.5

For values of factors (all four factors) that corresponds to a known passing style, the passing style is displayed alongside with it properties (name and factors). For example, values that corresponds to call by value (1, for all four selected factors), the passing style is displayed alongside it properties (name and factors). This can be seen below.

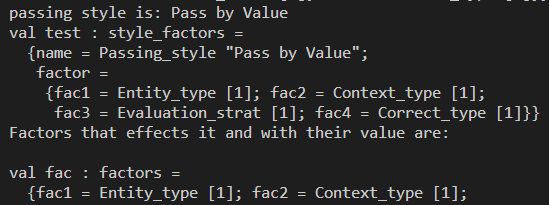


Fig.5. Known passing style

For the effects induced by values of factors for this passing style, used a record to hold the effects that the specific values of factors induces and is used to showcase this effects. It ask for a number which is used to initialize a variable init\_var, which is the initial value of the variable, eval is the result of a computation and final\_var is the final value of the variable after the computation. For call by value, we see that the value entered by user for the variable init\_var does not change as both values (init\_var and final\_var) are the same after performing a computation and not different form the value the user entered. This can be seen as shown below.

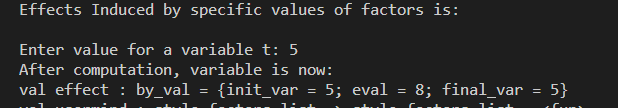


Fig.6

For adding/defining a new passing style (when values of selected factor(s) does not correspond to that of known passing style (e.g. 8, 8, for any two factors), new style is constructed and added to list of styles, and users then asked if they want more passing styles. If users go for no, the list of passing style is displayed which contain the newly added style. This can be seen below.

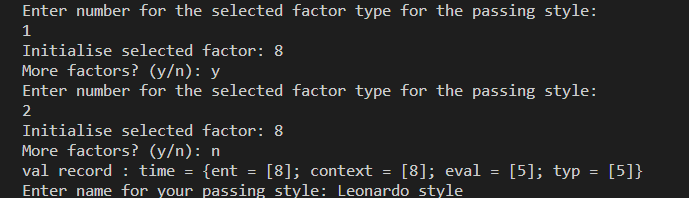


Fig.7. Defining new passing style

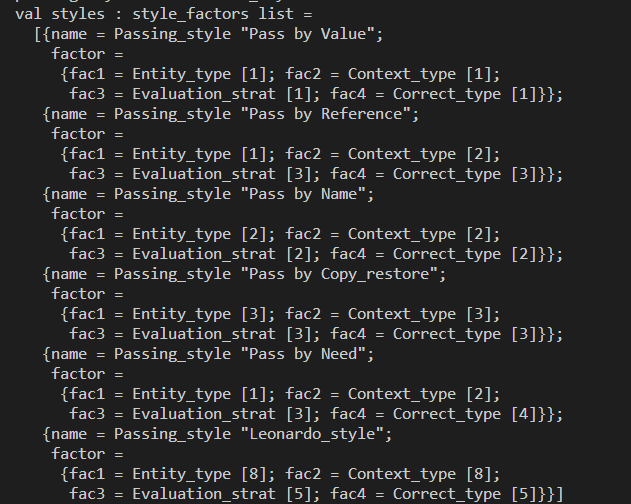


Fig.8. List containing newly added passing style.

Also for newly defined passing style, the style is displayed with it properties (name and factors) after has been defined. This can be seen as shown below.

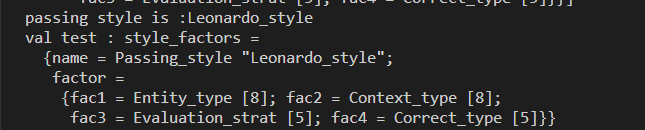


Fig.9. newly defined style.

After this process, if users wants to still carry out operations like add or remove a passing style, the method is invoked by typing the command (calling the method) “usermind styles” where usermind is the method invoked and styles refers to the current list of passing styles and selecting yes for more passing styles then select the operation they wish to carryout.

The method responsible for users selecting factors and initializing them can be seen below on the figure (Fig.10.) below, though not complete because of screen size issue.

Also the method used to ask users if they want more passing style (as in Fig.5) can be seen on the figure (Fig.11) below.

The methods “insert\_new\_passing\_style” and “removal” carries out operations for inserting and removing a passing style from the list of passing styles respectively.

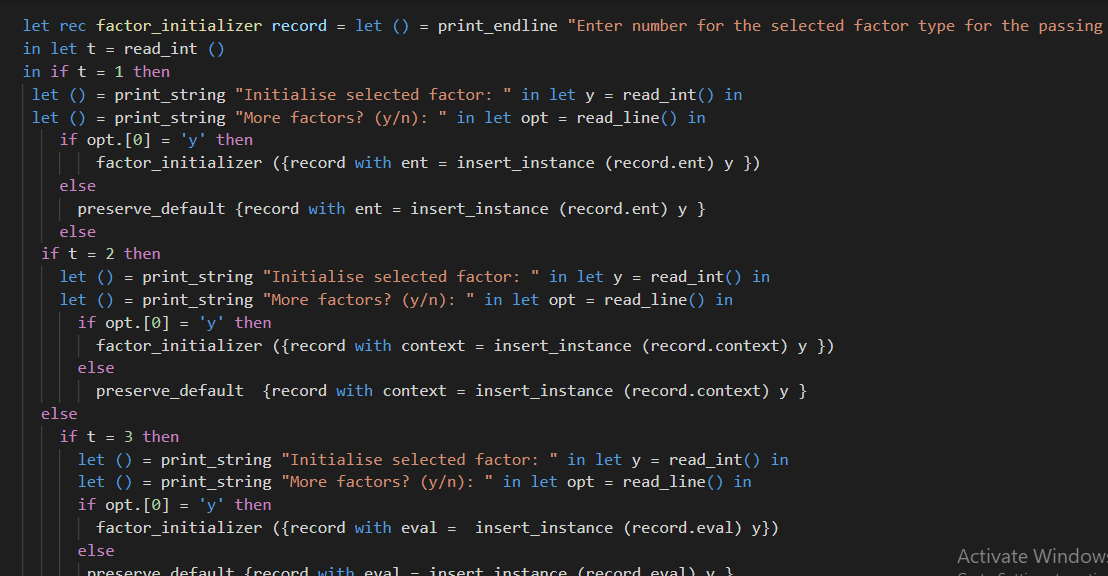


Fig.10

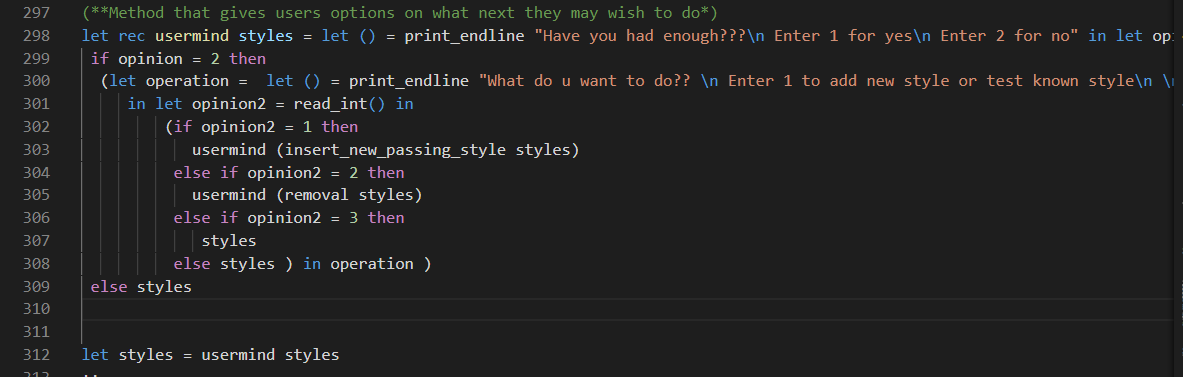


Fig.11