JAVA BYTE CODE GENERATOR

(Compiler)

Muhammed Essam (64)
Mohamed Murad(66)
Mohamed Raafat (62)
Wessam Mohamed (81)

Project Report

TABLE OF CONTENTS

1] USED DATA STRUCTURE

- Parser.y:
    - We only used an unordered map to map each variable name to a pair of integer and type which define its order in appearance and its type if is it float or integer.
    - We also used true_list to save the places of the true of Boolean expression to back patch it later, also we used the false_list for the same reason but for the falses of Boolean expressions, in addition we used next_list to save the next of each statement.

- Operation_mapper.h:
    - We used unordered_map to map each operation to its corresponding java byte code.

2] ALGORITHMS AND TECHNIQUES USED

- Conditional expressions:
    - in Boolean expressions we first generate two labels for true and false after that we send them to the expression. then the two labels are printed in front of the two statements responsible of true and false.

- Boolean expressions:
    - in Boolean expressions we generate the appropriate branch code using the inherited true and false attributes.

- For Loop:
    - In the loop we create the begin label after applying the first third of the for loop in order to begin the iteration then we process the Boolean expression as above, then we apply the loop_increment code and back patch its next with the begin label we created up, then we apply the body of the loop and back patch its next to the begin of the loop_increment part to apply the increament and loop again.

3] COMMENTS ABOUT THE TOOLS

- the tools are easy to use i found tutorails on them on the website, but there is no much tutorials or videos on the second part of using the jasmin to apply the java byte code, and for the generation of the java byte code itself using the bison also i see that this part in the leactures was not clarified clearly, but over all the tools are awesome.

4] EXPLANATION OF USED FUNCTION

- Write_code:
    - used to add the line of code to a vector in order to print the results to the terminal or to a file.

  Merge_list:
    - Used to merge the 2 lists
- Back_patch:
    - Used to apply the back patching to the gotos the doesn't have labels yet during the execution to make the complier compile the code in one pass.
- generate_label:
    - Take a number and generate the corresponding label string.
- Generate_header_java:
    - Header code must be added, that write the java byte code of the java class the definition of the main functions, as we only take care of the code inside the main.
- Generate_footer_jave:
    - Write the footer of the java byte code, also something must be added.

5] ANY ASSUMPTIONS MADE AND THEIR JUSTIFICATION

- Declaring and assigning values must be done on separate line.
- Assignment is done to the variables have the same type.