Birla Institute of Technology & Science-Pilani, Hyderabad Campus

First Semester 2017-2018

Principles of Programming languages (CS F301)

Assignment 2: Type Checking **Due Date: 15/11/2017**

Maximum Marks : 20

Goal and motivation:

This assignment aims at testing your understanding on type equivalence and type checking (Name, Internal Name and Structural).

Problem Statement:

Part1:

Write a code in C/C++/Java to implement the rules of Name, Internal Name and Structural Equivalence using the methods told in the class. Please refer to slides on Data Types.

1. Your code should include the rules for name equivalence checking. The brief rules are mentioned below. Refer to the lecture notes for detailed rules.
   - Types must have the exact same name to be equivalent
2. Your code should include the rules for Internal name equivalence checking. The brief rules are mentioned below. Refer to the lecture notes for detailed rules.
   - If the program interpreter gives the same internal name to two different variables, then they share the same type
3. Your code should include all the five rules for structural equivalence and the table entry manner to check for structural equivalence. Here is a short description of the five rules. Refer to the lecture notes for a detailed explanation.
   A. Same built in data types are structurally equivalent.
   B. Pointers to structurally equivalent types are structurally equivalent.
   C. Determining struct structural equivalence

a. Two structures
b. st1 { x_1: W_1, x_2: W_2, …, x_k: W_k }
c. st2 { y_1: Q_1, y_2: Q_2, ..., y_k: Q_k }
d. st1 and st2 are structurally equivalent iff
    i. W_1 structurally equivalent to Q_1
    ii. W_2 structurally equivalent to Q_2
    iii. ...
    iv. W_k structurally equivalent to Q_k

D. Determining array structural equivalence
    a. Two Arrays
    b. T1 = array range1 of t_1
    c. T2 = array range2 of t_2
    d. T1 and T2 are structurally equivalent iff:
        i. range1 and range2 have (1) the same number of dimensions and (2) the same number of entries in each dimension
        ii. t_1 and t_2 are structurally equivalent

E. Determining function structural equivalence
    a. Two functions
    b. T1 = function of (t_1, t_2, t_3, …, t_k) returns t
    c. T2 = function of (v_1, v_2, v_3, ..., v_k) returns v
    d. T1 and T2 are structurally equivalent iff:
        i. For all i from 1 to k, t_i is structurally equivalent to v_i
        ii. t is structurally equivalent to v

Use the structural equivalence algorithm as discussed in class (By constructing tables till T(N)= T(N-1)) to determine which two types are structurally equivalent.

Part2:

For the language you have chosen in Assignment 1, conduct a literature review of when and where name and structural equivalence is used. In the absence of such rules for name and structural equivalence formulate your own rules and submit the report.

For eg: In a language called DisneyLand (assuming this is the language I took in assignment-1),

The rules defined are as follows(Which you are expected define by yourself for your language if your language does not have datatypes then you can define it for C langauge):

1. Name equivalence is used for basic data types.
2. Structural equivalence is used for structures.
3. Name equivalence is used for arrays.

So, you are expected to write a code which checks all of the three rules for your language using the name and structural equivalence rules in Part1.

Eg:

A code snippet in DisneyLand language is as follows:

Testcase1:

```
int a,b;
int c;
float x,y;
float z;
int array[0-4] q,r;
int array[0-4] s,
struct foo{
int h,
string i;
};
struct cat{
int d;
string f;
};
Struct foo aa,bb;
Struct cat cc;
//similarly for functions as well.
```

Output Format for Testcase1 from Code 1

Table to determine structural Equivalence-

	a	b	c	x	y	z	q	r	s	aa	bb	cc
a	Y	Y	Y	N	N	N	N	N	N	N	N	N
b		Y	Y	N	N	N	N	N	N	N	N	N
c			Y	N	N	N	N	N	N	N	N	N
x				Y	Y	Y	N	N	N	N	N	N
y					Y	Y	N	N	N	N	N	N
z						Y	N	N	N	N	N	N
q							Y	Y	Y	N	N	N
r								Y	Y	N	N	N
s									Y	N	N	N
aa										Y	Y	Y
bb											Y	Y
cc												Y

Name and Internal Name Equivalence

1) a,b,c are name and internal name equivalent.
2) x,y,z are nameand internal name equivalent.
3) q,r are name equivalent.q and s are internal name equivalent. r and s are internal name equivalent.
4) aa and bb are name equivalent.

For **'TestCase-1'**, your code for Part-1 should print the output as shown above. And when you run your code 2, the user is expected to enter an input in the format specified below.

Output for Testcase1 code in part-2

$ gcc part-2.c

$.\a.out

a,b ***User-Input***

True as a,b are name equivalent \\ **Your code output**

q,s

False as q,s are not name equivalent \\ **Your code output**

aa,cc

True as aa,cc are structurally equivalent \\ **Your code output**

Step by step explanation of how your code should work:

Please note that the output of part-1 and the rules specific to your grammar will determine your output in part-2.

The codes are expected to work in this way.

For a specified input file with the variable declarations, your code for part-1 should print the structural equivalence table and name equivalence and internal name equivalence results.

Your code two should use the output of code-1 and according to the rules of language, should say if the variables user entered are equivalent.
For example, when user enters a, b when code-2 is run on the terminal, your code-2 checks

1) What is the type of a and b which says a and b are integers

2) Now, we consult the rules for integers and know that two integers are equivalent only if they are name equivalent.

3) We go to the output of code-1 and check if a and b are name equivalent. If they are you need to print true, else false stating the reason as mentioned above.

Deliverables:

1. Code for part1

2. Code for part2

3. Rules for part 2

4. The test case and their output tables and print all the pairs of variables that are Internal Name, Name and Structural Equivalent for each test case (Output of code-1) and during the demo your code will be expected to answer to the user input based on your code like shown above. (Output of code-2)

5. Readme.txt file specifying the execution process of the codes and input and output folders

Submissions:

The assignment has to be submitted before the due date (11.59 p.m. on 15/11/2017) to

email id:- ppltwenty.17@gmail.com.

The subject of email should be "Assignment-2 First Sem 2017-18".

Please make sure that the body of the email has the names and id numbers of all the students in the group. Name the assignment zip folder as Asst2_id1_id2_id3. (Eg. Asst2_14005_14096_14130.tar.gz)