ROUND – 2

Marking Scheme

- Input source code will be considered as a long string (length *‘n’*) and will be broken into non overlapping substrings of length 13 each and last substring of length *n modulo 13*.

- Score of a substring will be calculated using the function *foo1*.

```
Function foo1(string s)
  w = [-6,-5,-4,-3,-2,-1,0,1,2,3,4,5,6]
  substring_score = 0
  for i = 1..(s.length)
      substring_score += ascii(s[i]) * w[i]
  return substring_score
```

- Function *foo2* computes the score for all the substrings in the program P.

```
Function foo2(Program P)
    substrings = [all substrings of P]
    total_score = 0
    for i = 1..substrings.size
          total_score += foo1(substrings[i])
    return total_score
```

- Function *finalscore* computes the final score of the problem out of 100

```
sum(P) :- 3 * 122 * P.length;

Function final(Program P)
    total_score = foo2(P)
    fractional_score = total_score/sum(P)
    final_score = 100*fractional_score
    return final_score
```

Example :

```
code = “””
           int i = 10;
           i = 1;
       “””
```

2 substrings are :

- {‘i’, ‘n’, ‘t’, ‘ ‘ , ‘i’ , ‘ ‘ , ‘=’ , ‘ ‘ , ‘1’ , ‘0’, ‘;’ , ‘\n’ , ‘i’}

- {‘ ‘ , ‘=’ , ‘ ‘ , ‘1’ , ‘;’}

```
ascii(‘i’) = 105    ascii(‘n’) = 110
ascii(‘t’) = 116    ascii(‘ ‘) = 32
ascii(‘=’) = 61     ascii(‘1’) = 49
ascii(‘0’) = 48     ascii(‘;’) = 59
ascii(‘\n’) = 13
```

Score of substring 1 :

total_score = 105 * (-6) + 110 * (-5) + 116 * (-4) + 32 * (-3) + 105 * (-2) + 32 * (-1) + 61 * (0) + 32 * (1) + 49 * (2) + 48 * (3) + 59 * (4) + 13 * (5) + 105 * (6) = -777

*final_score = -777/(3*122*18) * 100 = -11.79*

NOTE : Space/Tab as well as Newline character will be included for score computation. You may get negative scores for submission as well therefore be very careful before submitting.

RANKING ORDER : Preference given to total points scored, if equal then number of problems solved and if equal then total attempts made.