***ASSIGNMENT-2***

***QUESTION NO.:-3***

***ALGORITHM:-***

*CHANGE (n)*

Statement: This function take a number ‘n’ as argument.

Step 1: Return the value of (n-48) //here we subtract 48 from ‘n’ to get exact value of the number ASCII number

Step 2: END

*CREATE (num)*

Statement: This function take a string of the number which is define by ‘num’.

Step 1: Initialize ‘i’ by 0

Step 2: Repeat step 3 while (num[i]≠EOF) do,

Step 3: Increase the value of ‘i’ by 1

Step 4: Repeat step 5 to 6 while (i≥0) do,

Step 5: num[i+1]←num[i]

Step 6: Decrement the value of ‘i’ by 1

Step 7: Initialize ‘num[0]’ by 0.

Step 8: END

*MAIN ()*

Step 1: Initialize ‘k’ by 0.

Step 2: Input the string of number in ‘num’.

Step 3: Input the accuracy of square root of the number in ‘acc’.

Step 4: len←calculate the length of string ‘num’ by using the library function ‘strlen’.

Step 5: If (len%2≠0) then go to step 6,

Step 6: CREATE(num) //call the CREATE function if length of string is odd

Step 7: Increase the value of ‘len’ by 1

Step 8: Initialize ‘r’ and ‘div’ by 0

Step 9: Initialize ‘i’ by 0

Step 10: Repeat step 11 to 26 while (i<(len/2)) do,

Step 11: c1←CHANGE(num[i\*2]) //call the CHANGE function

Step 12: c2←CHANGE(num[(i\*2)+1]) //call the CHANGE function

Step 13: d←(c1\*10)+c2

Step 14: d←(r\*100)+d

Step 15: Initialize ‘j’ by 1

Step 16: t←((div\*2)\*10)+j

Step 17: Repeat step 18 to 19 while (t\*j≤d) do,

Step 18: j←j+1

Step 19: t←((div\*2)\*10)+j

Step 20: j←j-1

Step 21: t←((div\*2)\*10)+j

Step 22: n[k]←(j+48) //here we add 48 with ‘j’ to get the ASCII number of ‘j’

Step 23: k←k+1

Step 24: r←d-(t\*j)

Step 25: div←(div\*10)+j

Step 26: i←i+1

Step 27: If (r≠0) the go to step 28 to 53,

Step 28: n[k]←’.’

Step 29: k←k+1

Step 30: Initialize ‘i’ by 0

Step 31: Repeat step 32 to 33 while (i<(acc\*2)) do,

Step 32: p[i]←put the ASCII value of 0

Step 33: i←i+1

Step 34: p[i]←’\0’ [// ‘\0](file:///\\'\0)’ means NULL which behave like end of string

Step 35: len←acc\*2

Step 36: Initialize ‘i’ by 0

Step 37: Repeat step 38 to 53 while (i<(len/2)) do,

Step 38: c1←CHANGE(num[i\*2]) //call the CHANGE function

Step 39: c2←CHANGE(num[(i\*2)+1]) //call the CHANGE function

Step 40: d←(c1\*10)+c2

Step 41: d←(r\*100)+d

Step 42: Initialize ‘j’ by 1

Step 43: t←((div\*2)\*10)+j

Step 44: Repeat step 45 to 46 while (t\*j≤d) do,

Step 45: j←j+1

Step 46: t←((div\*2)\*10)+j

Step 47: j←j-1

Step 48: t←((div\*2)\*10)+j

Step 49: n[k]←(j+48) //here we add 48 with ‘j’ to get the ASCII number of ‘j’

Step 50: k←k+1

Step 51: r←d-(t\*j)

Step 52: div←(div\*10)+j

Step 53: i←i+1

Step 54: n[k]←’\0’ [// ‘\0](file:///\\'\0)’ means NULL which behave like end of string

Step 55: Display the value of the array ‘n’ which store the square root of number

Step 56: END