Data Structures : Session 1 (02/07/2020)

Question 1:

Need of Data Type in C?

A DataType in programming language, is a classification that specifies which type of value a variable has and what type of mathematical, relational or logical operations can be performed on those variables. Data Type allots the significant amount of memory which is required for the specific variable. It also help to specify what type of data is required from the user.

Question 2:

Error of %s ?

The error occurred due to the spaces in “Let us C” statement. C considers “ “(space) as null “/0” so it stops reading or scanning the parts after spaces. T scan a statement with spaces we can use gets() function.

Question 3:

Code : Creation of student structure?

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct Student

{

int ID;

char name[20];

int Year;

float CGPA;

};

int main()

{

struct Student s1;

s1.ID = 1191;

strcpy(s1.name, "Rutika");

s1.Year = 1;

s1.CGPA = 9.2;

printf("Student 1: ID - %d \n Name - %s \n Year - %d \n CGPA - %f \n\n", s1.ID, s1.name, s1.Year, s1.CGPA);

struct Student s2;

s2.ID = 1191;

strcpy(s2.name, "Rohan");

s2.Year = 3;

s2.CGPA = 9.0;

printf("Student 2: ID - %d \n Name - %s \n Year - %d \n CGPA - %f \n\n", s2.ID, s2.name, s2.Year, s2.CGPA);

struct Student s3;

s3.ID = 1191;

strcpy(s3.name, "Saee");

s3.Year = 2;

s3.CGPA = 8.3;

printf("Student 3: ID - %d \n Name - %s \n Year - %d \n CGPA - %f \n\n", s3.ID, s3.name, s3.Year, s3.CGPA);

struct Student s4;

s4.ID = 1191;

strcpy(s4.name, "Gauravi");

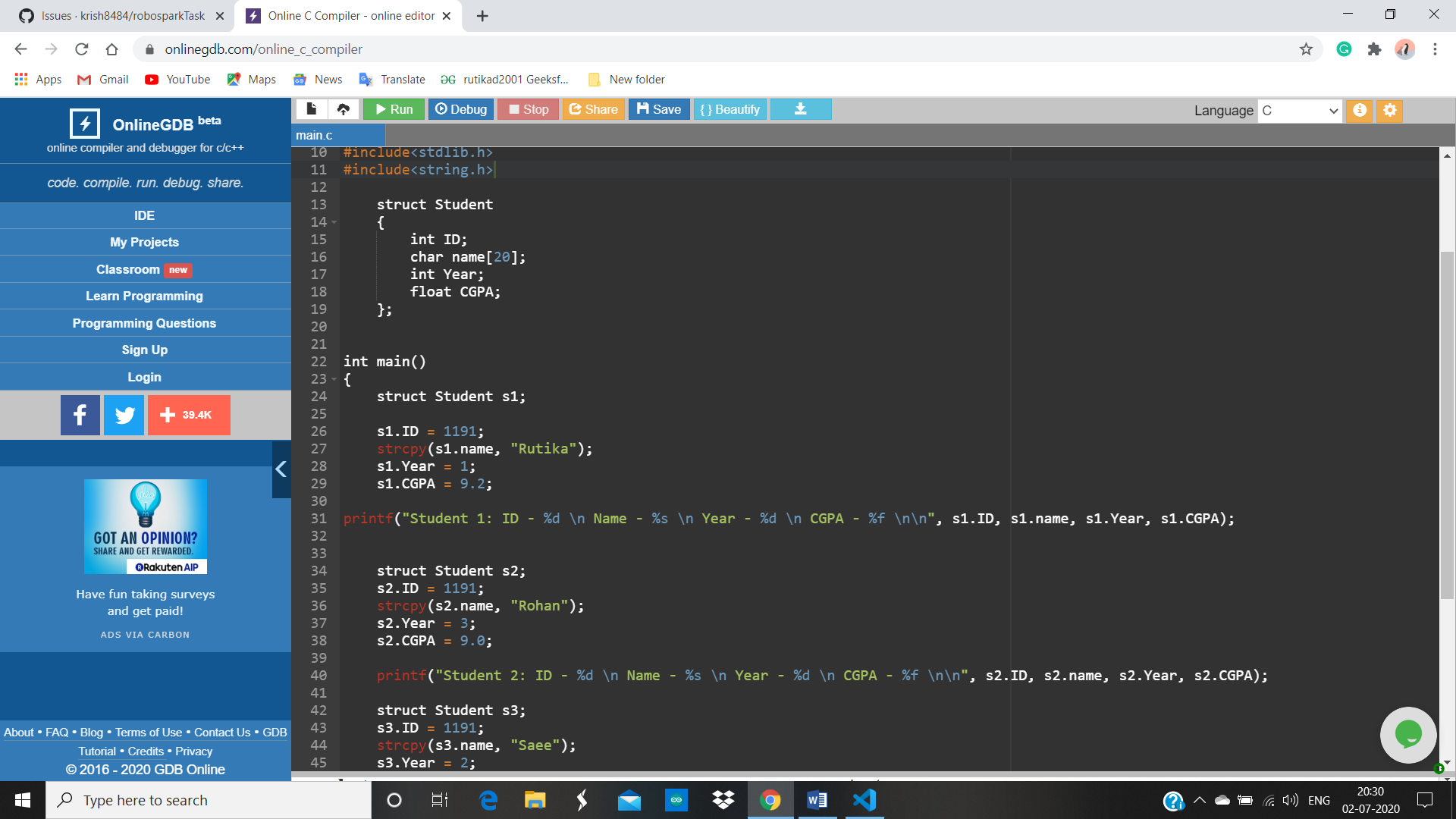
s4.Year = 4;

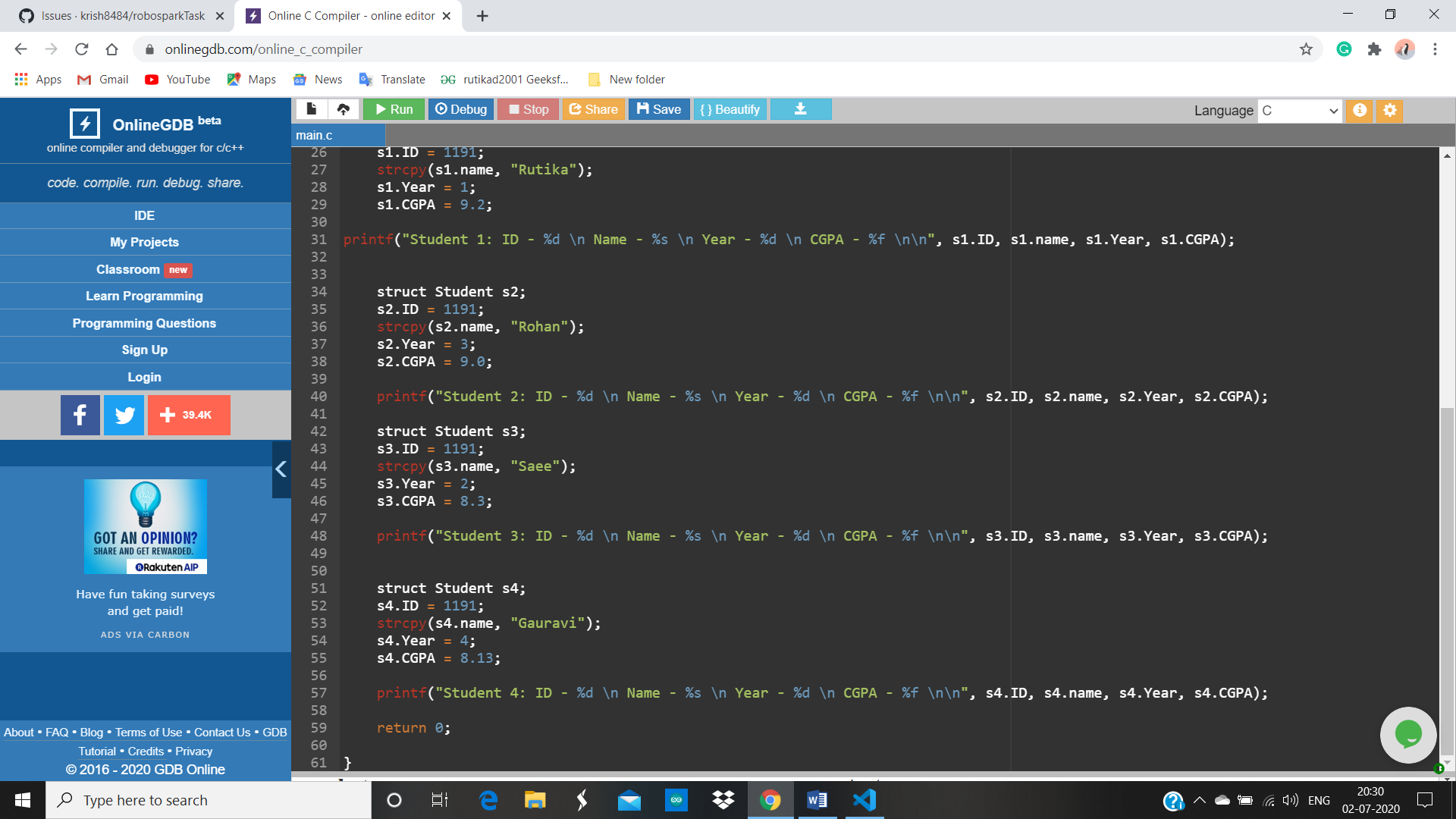
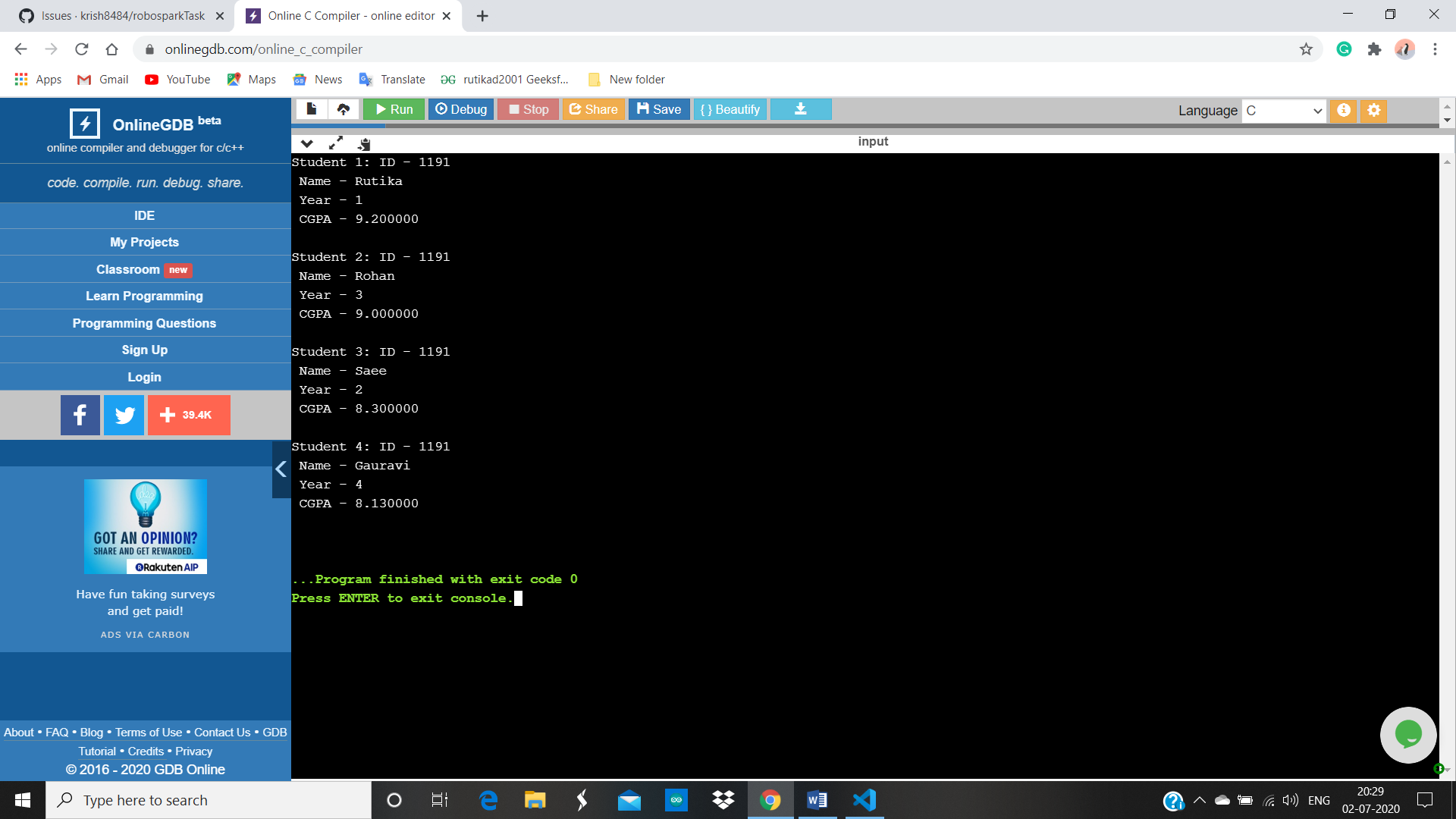
s4.CGPA = 8.13;

printf("Student 4: ID - %d \n Name - %s \n Year - %d \n CGPA - %f \n\n", s4.ID, s4.name, s4.Year, s4.CGPA);

return 0;

}

Screenshots: 



Question 4:

Balance bracket problem:

Code:

#include<stdio.h>

int push(char element, int \*stack, int top)

{

top = top +1 ;

stack[top] = element;

return top;

}

int pop( int top)

{

top = top-1;

return top;

}

int isRev(char element, int \*stack, int top)

{

if (stack[top]=="(")

{

if (element == ")")

{

return 1;

}

else

{

return 0;

}

}

else if (stack[top]=="{")

{

if (element == "}")

{

return 1;

}

else

{

return 0;

}

}

else if (stack[top]=="[")

{

if (element == "]")

{

return 1;

}

else

{

return 0;

}

}

else

{

return 0;

}

}

int main()

{

char stack[10], string[10];

int top = -1;

printf("Enter a string of brackets :");

scanf("%s", string);

int i =0;

while (string[i]!='\0')

{

if(isRev(string[i], stack, top) && top >= 0)

{

top = pop(top);

}

else

{

top = push(string[i], stack, top);

}

i = i+1;

}

if( top == -1)

{

printf("The brackets are valid !!!");

}

else

{

printf("invalid Brackets !!!");

}

return 0;

}

Screenshot: 