

Ensenada, Baja California a 05 de octubre del 2020



Practica No.4

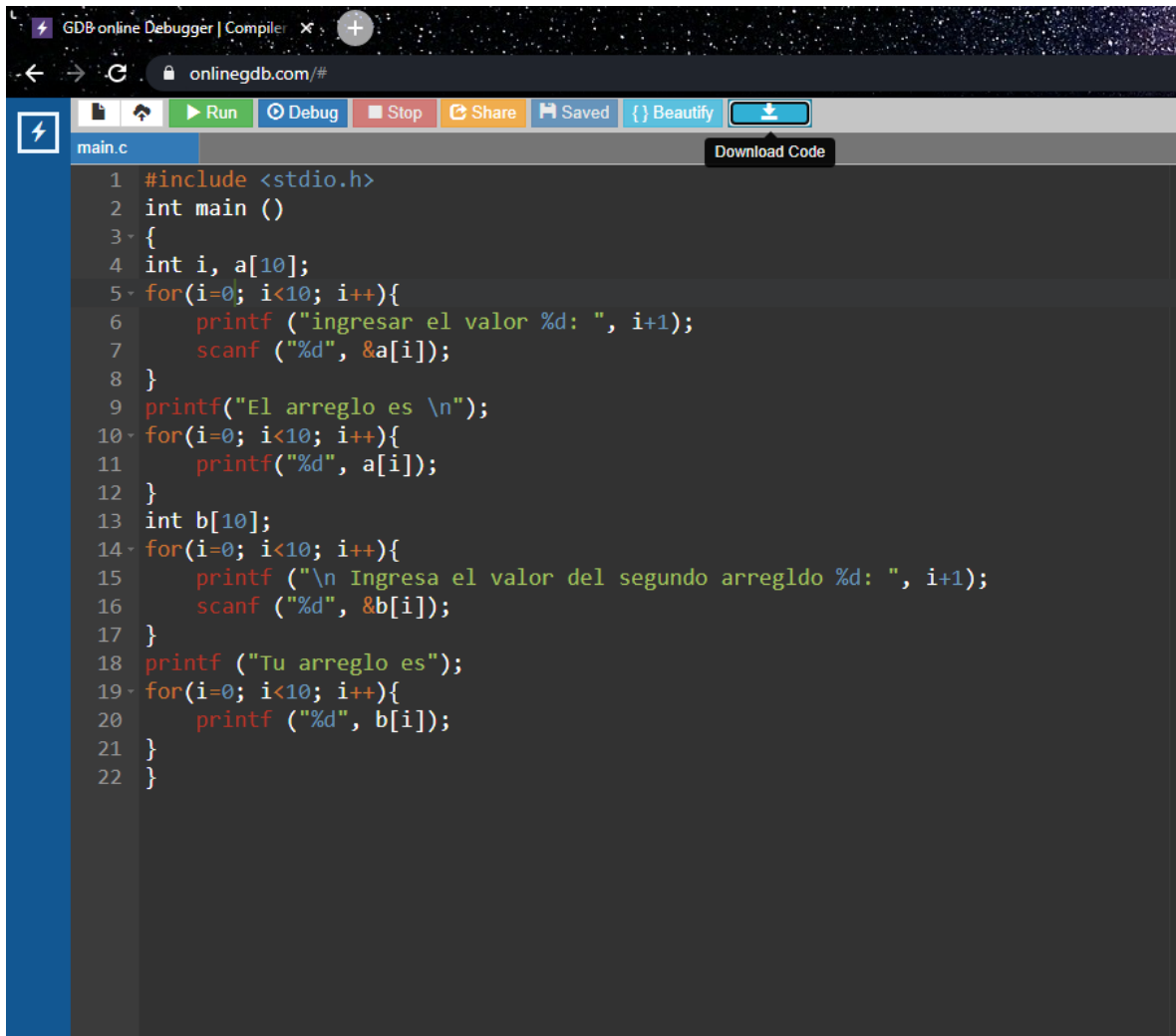
Funciones, arreglos y matrices

Profesor: Roilhi Frajo Ibarra Hernández

Alumno: Fabian Diaz Fajardo

Grupo: 021

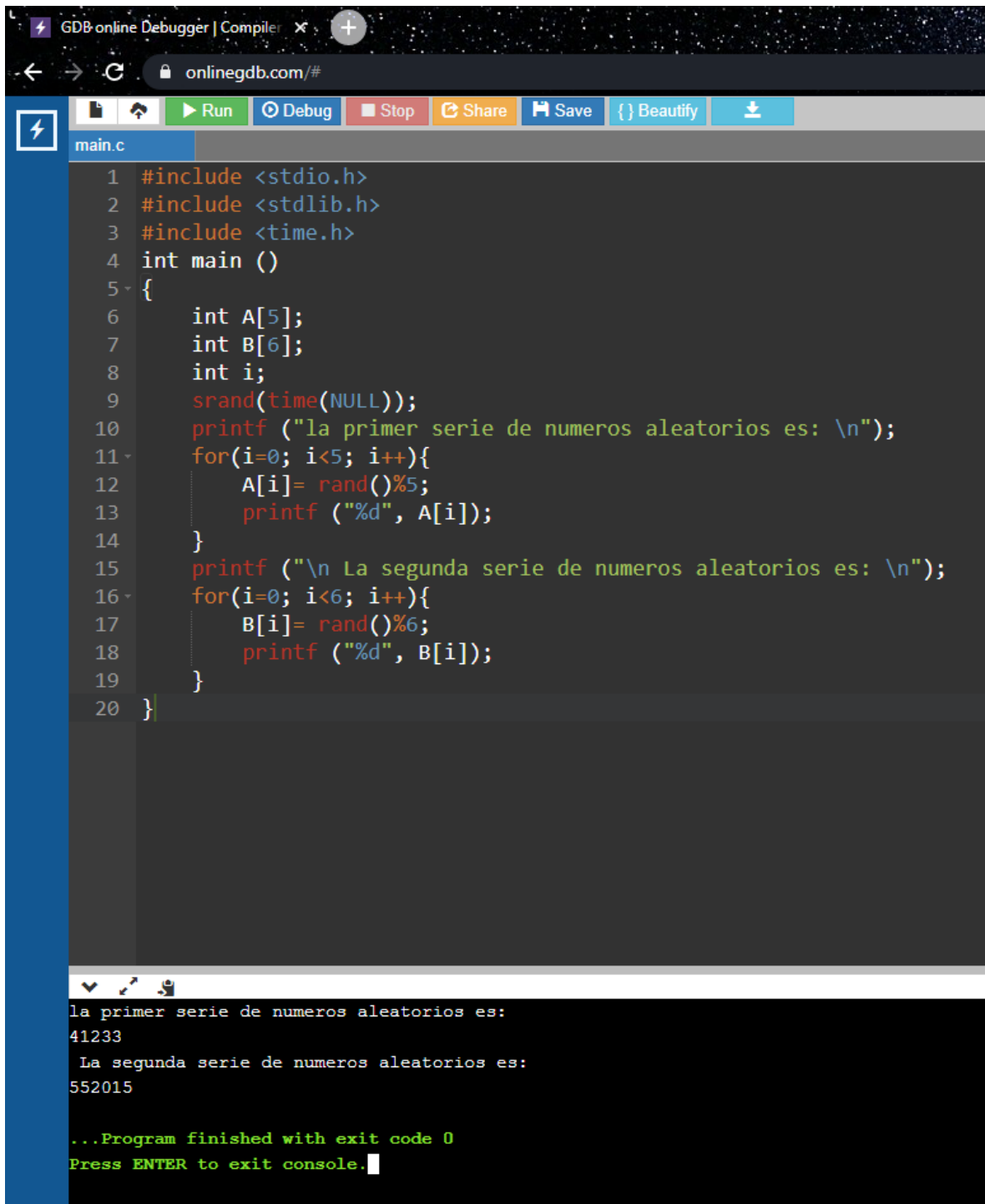
4.1.- El usuario va a ingresar 2 arreglos vectores elemento a elemento. Usar un tamaño de 6 a 10 para ambos elementos. Si se Deberá guardarse esta operación dentro de una función. La función deberá tener la opción de imprimir los dos arreglos.



The screenshot shows the GDB online Debugger interface. The browser address bar displays 'onlinegdb.com/#'. The interface includes a toolbar with buttons for 'Run', 'Debug', 'Stop', 'Share', 'Saved', 'Beautify', and 'Download Code'. The file 'main.c' is open in the editor. The code is as follows:

```
1 #include <stdio.h>
2 int main ()
3 {
4     int i, a[10];
5     for(i=0; i<10; i++){
6         printf ("ingresar el valor %d: ", i+1);
7         scanf ("%d", &a[i]);
8     }
9     printf("El arreglo es \n");
10    for(i=0; i<10; i++){
11        printf("%d", a[i]);
12    }
13    int b[10];
14    for(i=0; i<10; i++){
15        printf ("\n Ingresa el valor del segundo arregldo %d: ", i+1);
16        scanf ("%d", &b[i]);
17    }
18    printf ("Tu arreglo es");
19    for(i=0; i<10; i++){
20        printf ("%d", b[i]);
21    }
22 }
```

4.2.- Se generarán 2 vectores con elementos enteros y se llenarán de forma aleatoria (verificar la función rand()). Guardar la operación dentro de una función e imprimir ambos arreglos.



The screenshot shows a web-based GDB compiler interface. The top bar includes a lightning bolt icon, the text 'GDB online Debugger | Compiler', and a close button. Below this is a navigation bar with icons for back, forward, refresh, and a URL bar showing 'onlinegdb.com/#'. A secondary toolbar contains buttons for 'Run' (green), 'Debug' (blue), 'Stop' (red), 'Share' (orange), 'Save' (blue), 'Beautify' (blue with curly braces), and a download icon. The main editor area, titled 'main.c', contains the following C code:

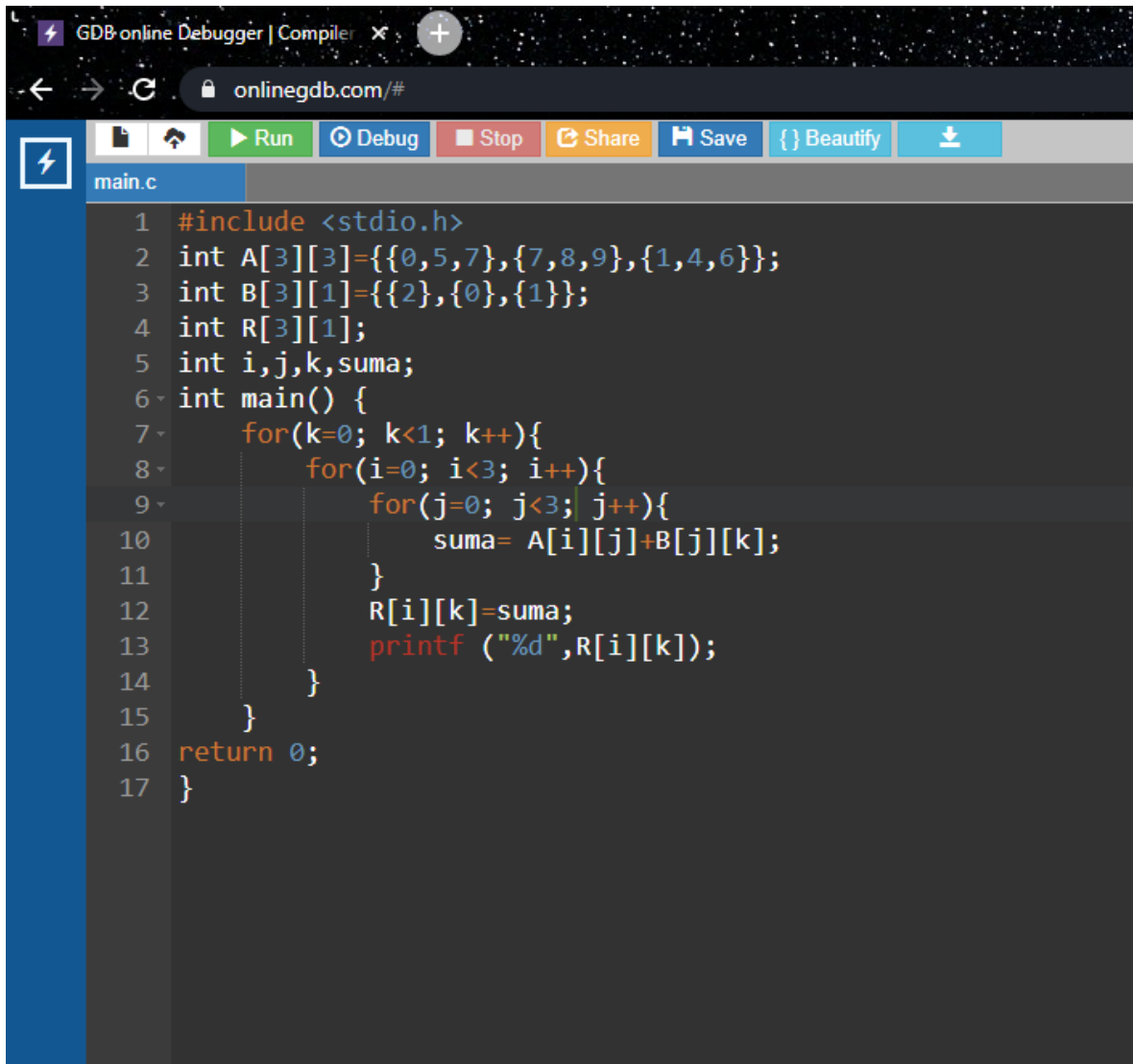
```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4 int main ()
5 {
6     int A[5];
7     int B[6];
8     int i;
9     srand(time(NULL));
10    printf ("la primer serie de numeros aleatorios es: \n");
11    for(i=0; i<5; i++){
12        A[i]= rand()%5;
13        printf ("%d", A[i]);
14    }
15    printf ("\n La segunda serie de numeros aleatorios es: \n");
16    for(i=0; i<6; i++){
17        B[i]= rand()%6;
18        printf ("%d", B[i]);
19    }
20 }
```

Below the code editor is a console window with a dark background. It displays the program's output:

```
la primer serie de numeros aleatorios es:
41233
La segunda serie de numeros aleatorios es:
552015

...Program finished with exit code 0
Press ENTER to exit console.
```

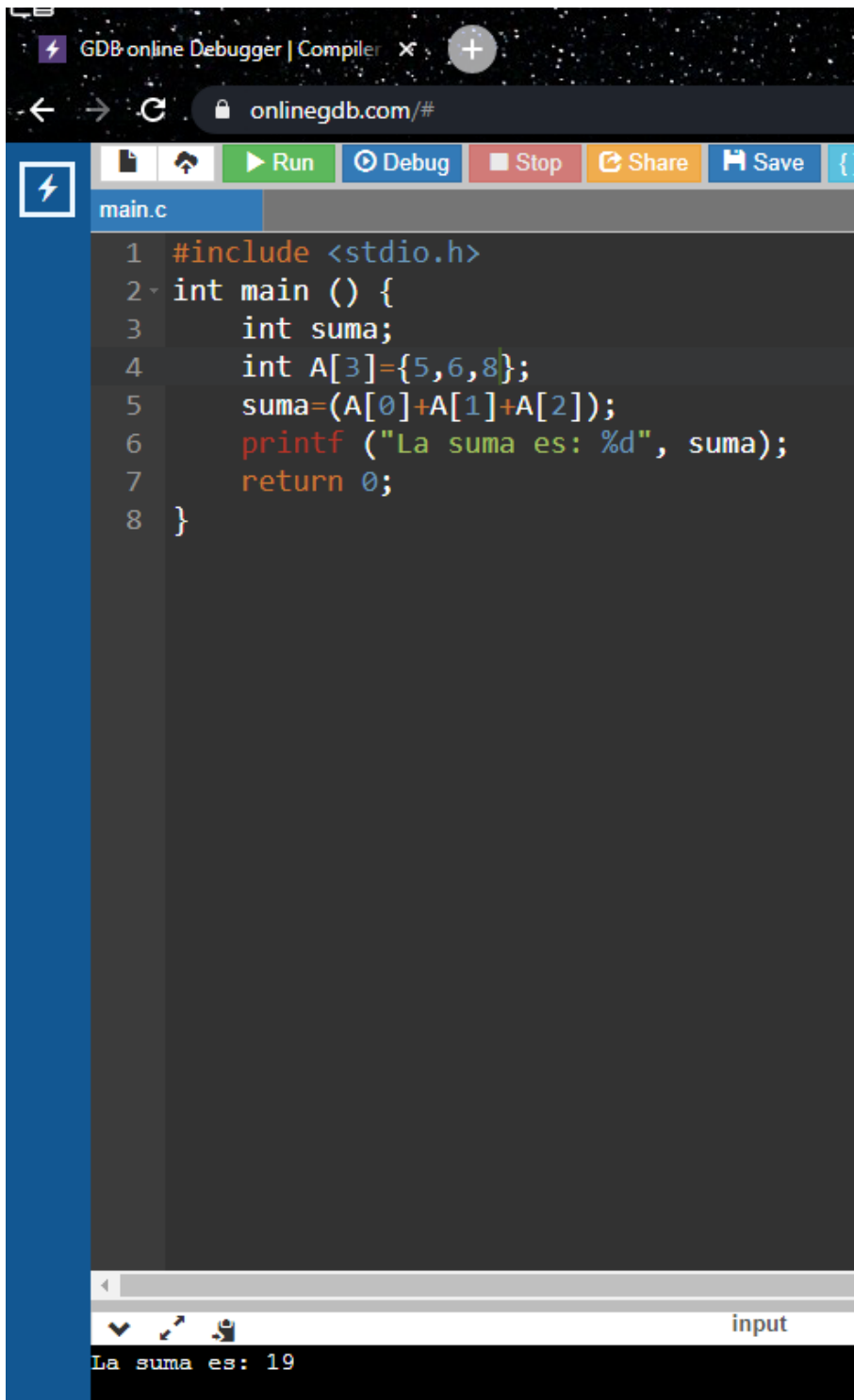
4.3.- Generar una función que realice la suma de los dos arreglos vectoriales e imprima el resultado a la salida.



The screenshot shows the GDB online Debugger interface. The browser address bar displays 'onlinegdb.com/#'. The interface includes a toolbar with buttons for 'Run', 'Debug', 'Stop', 'Share', 'Save', 'Beautify', and a download icon. The file 'main.c' is open in the editor. The code defines two 3x3 matrices, A and B, and a 3x3 matrix R. Matrix A is initialized with the values {{0,5,7},{7,8,9},{1,4,6}}. Matrix B is initialized with the values {{2},{0},{1}}. The program then iterates over the indices of matrix A (i from 0 to 2) and matrix B (j from 0 to 2), calculating the sum of the corresponding elements and storing the result in matrix R. The result is printed using printf.

```
1 #include <stdio.h>
2 int A[3][3]={{0,5,7},{7,8,9},{1,4,6}};
3 int B[3][1]={{2},{0},{1}};
4 int R[3][1];
5 int i,j,k,suma;
6 int main() {
7     for(k=0; k<1; k++){
8         for(i=0; i<3; i++){
9             for(j=0; j<3; j++){
10                suma= A[i][j]+B[j][k];
11            }
12            R[i][k]=suma;
13            printf ("%d",R[i][k]);
14        }
15    }
16    return 0;
17 }
```

4.4.- Generar una función que realice la suma de todos los elementos de un arreglo e imprima el resultado a la salida. Ejemplo: para el arreglo $a=[2\ 4\ 6]$ el resultado sería $2+4+6=12$.



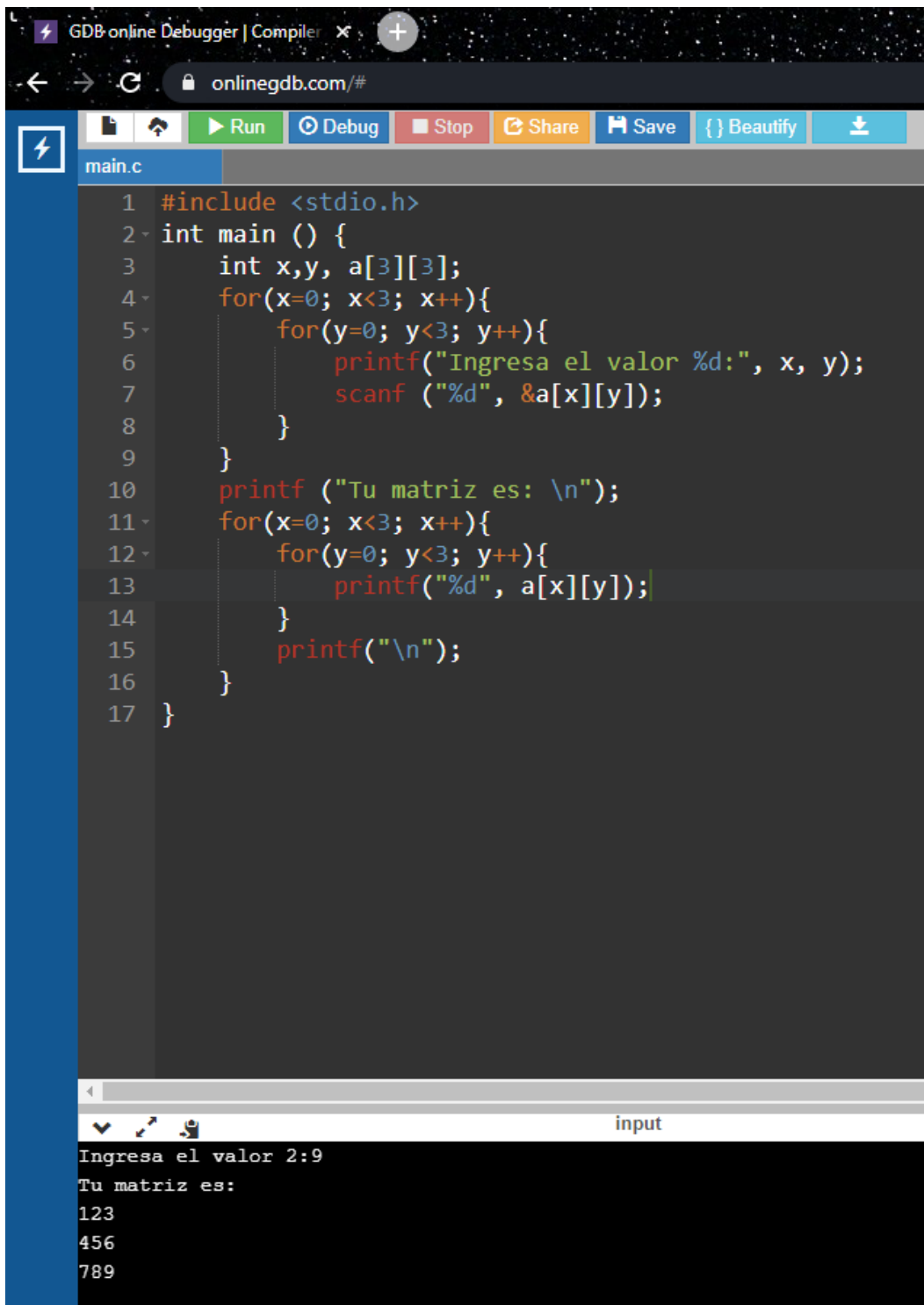
The screenshot shows the GDB online Debugger interface. The browser address bar displays "onlinegdb.com/#". The interface includes a toolbar with buttons for "Run", "Debug", "Stop", "Share", and "Save". The file "main.c" is open, showing the following C code:

```
1 #include <stdio.h>
2 int main () {
3     int suma;
4     int A[3]={5,6,8};
5     suma=(A[0]+A[1]+A[2]);
6     printf ("La suma es: %d", suma);
7     return 0;
8 }
```

At the bottom, the output window shows the result of the program execution:

```
La suma es: 19
```

4.5.- Tomar la función de llenado del arreglo vectorial por el usuario (parte 4.1) y extenderlo a dos dimensiones. Esto es, se creará una función donde el usuario va a llenar una matriz cuadrada (3x3 o 4x4).



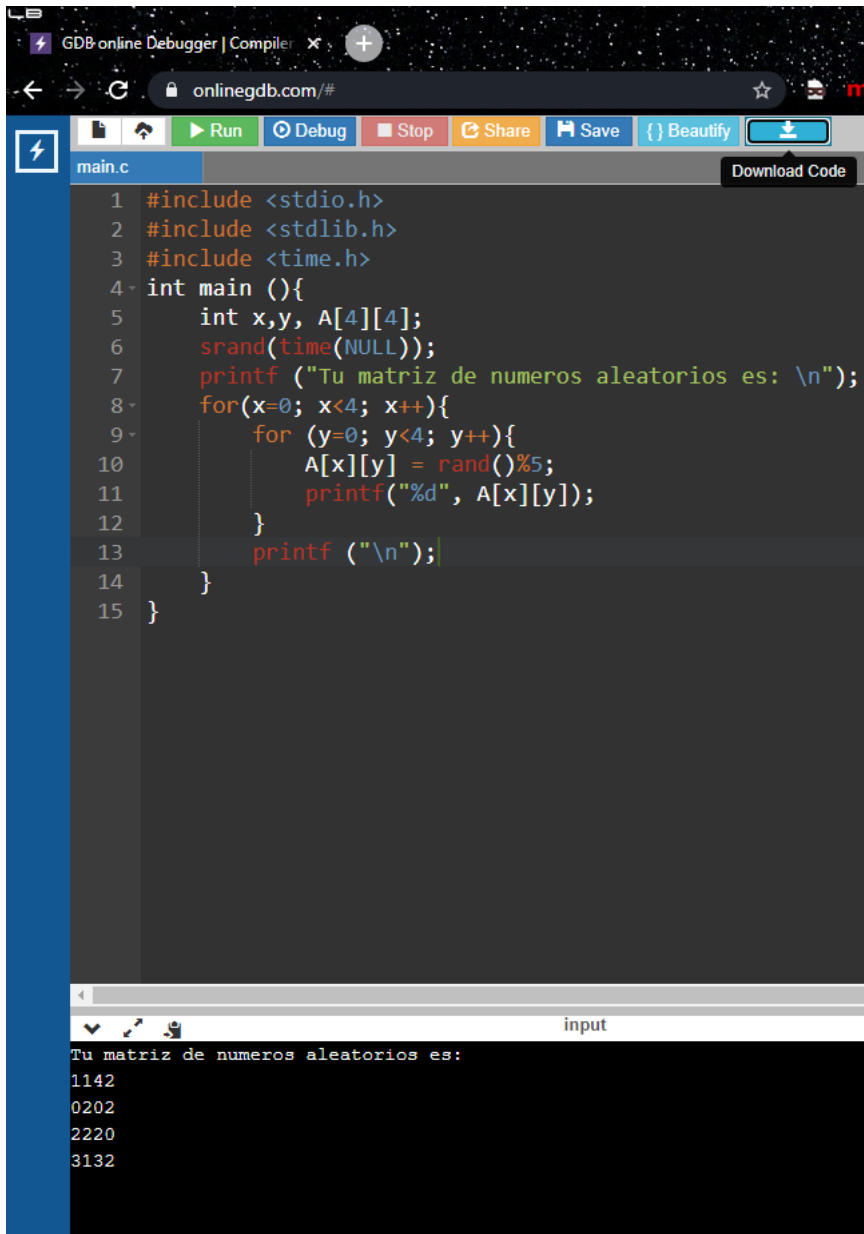
The image shows a screenshot of the GDB-online Debugger interface. The top bar includes the title 'GDB-online Debugger | Compiler', a search icon, and a plus sign. Below the bar is a navigation menu with icons for file, run, debug, stop, share, save, beautify, and download. The main editor displays a C program named 'main.c' with the following code:

```
1 #include <stdio.h>
2 int main () {
3     int x,y, a[3][3];
4     for(x=0; x<3; x++){
5         for(y=0; y<3; y++){
6             printf("Ingresa el valor %d:", x, y);
7             scanf ("%d", &a[x][y]);
8         }
9     }
10    printf ("Tu matriz es: \n");
11    for(x=0; x<3; x++){
12        for(y=0; y<3; y++){
13            printf("%d", a[x][y]);
14        }
15        printf("\n");
16    }
17 }
```

Below the code editor is a terminal window with the label 'input'. It shows the program's execution output:

```
Ingresa el valor 2:9
Tu matriz es:
123
456
789
```

4.6.- Tomar la función de llenado del arreglo vectorial con números aleatorios (parte 4.2) y extenderlo a dos dimensiones. Esto es, se creará una función donde se va a llenar una matriz cuadrada (3x3 o 4x4) con números enteros aleatorios.



The screenshot shows the onlinegdb.com web interface. The top navigation bar includes buttons for Run, Debug, Stop, Share, Save, Beautify, and Download Code. The main editor displays a C program named 'main.c' with the following code:

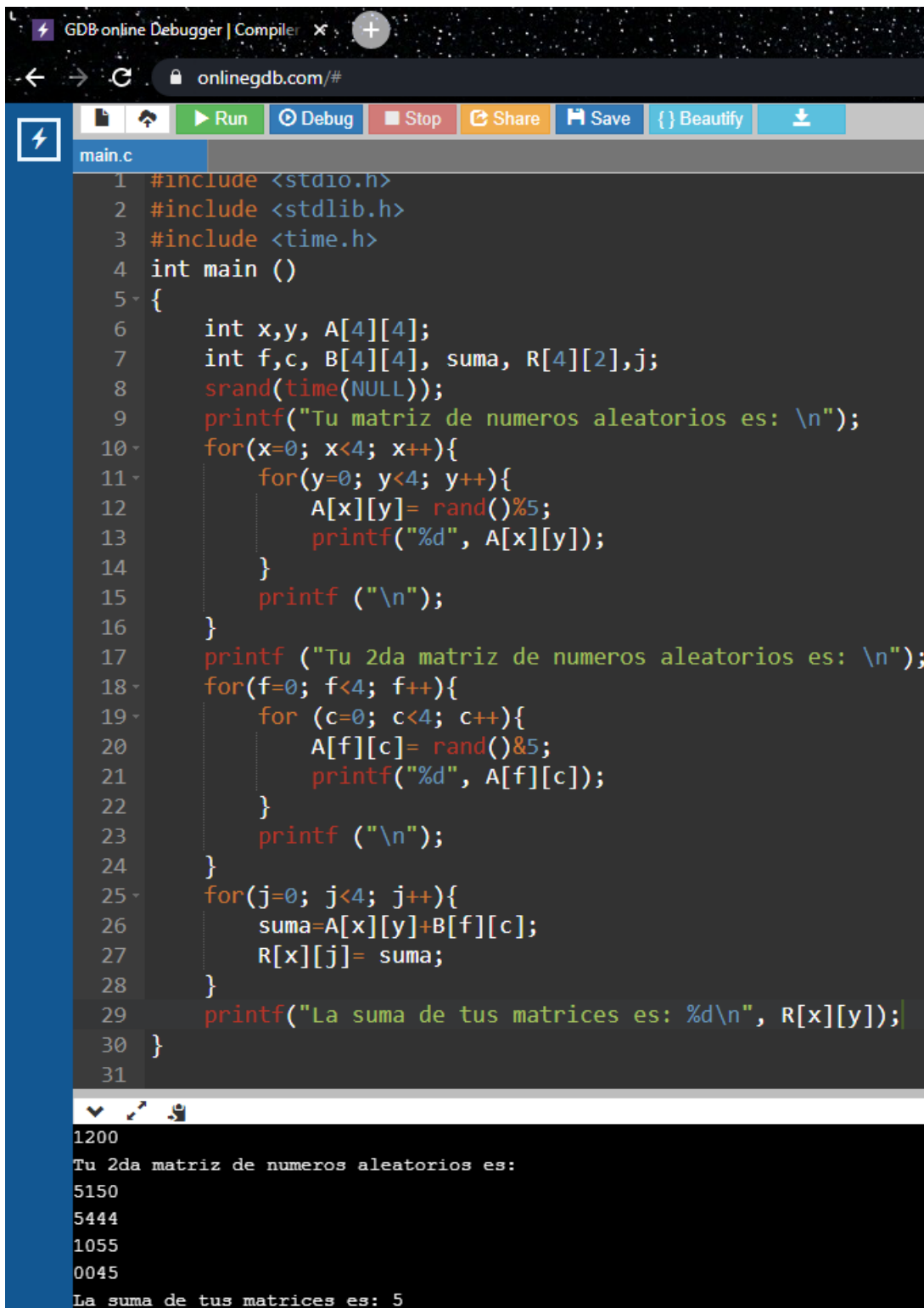
```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4 int main (){
5     int x,y, A[4][4];
6     srand(time(NULL));
7     printf ("Tu matriz de numeros aleatorios es: \n");
8     for(x=0; x<4; x++){
9         for (y=0; y<4; y++){
10             A[x][y] = rand()%5;
11             printf("%d", A[x][y]);
12         }
13         printf ("\n");
14     }
15 }
```

Below the code editor, the 'input' section shows the program's output:

```
Tu matriz de numeros aleatorios es:
1142
0202
2220
3132
```

The program has finished with exit code 0.

4.7.- Tomar la función de suma de arreglos vectoriales (parte 4.2) y extender la operación a dos dimensiones. Es decir, la función sumará las dos matrices o arreglos bidimensionales. Deberá imprimir el resultado a la salida.



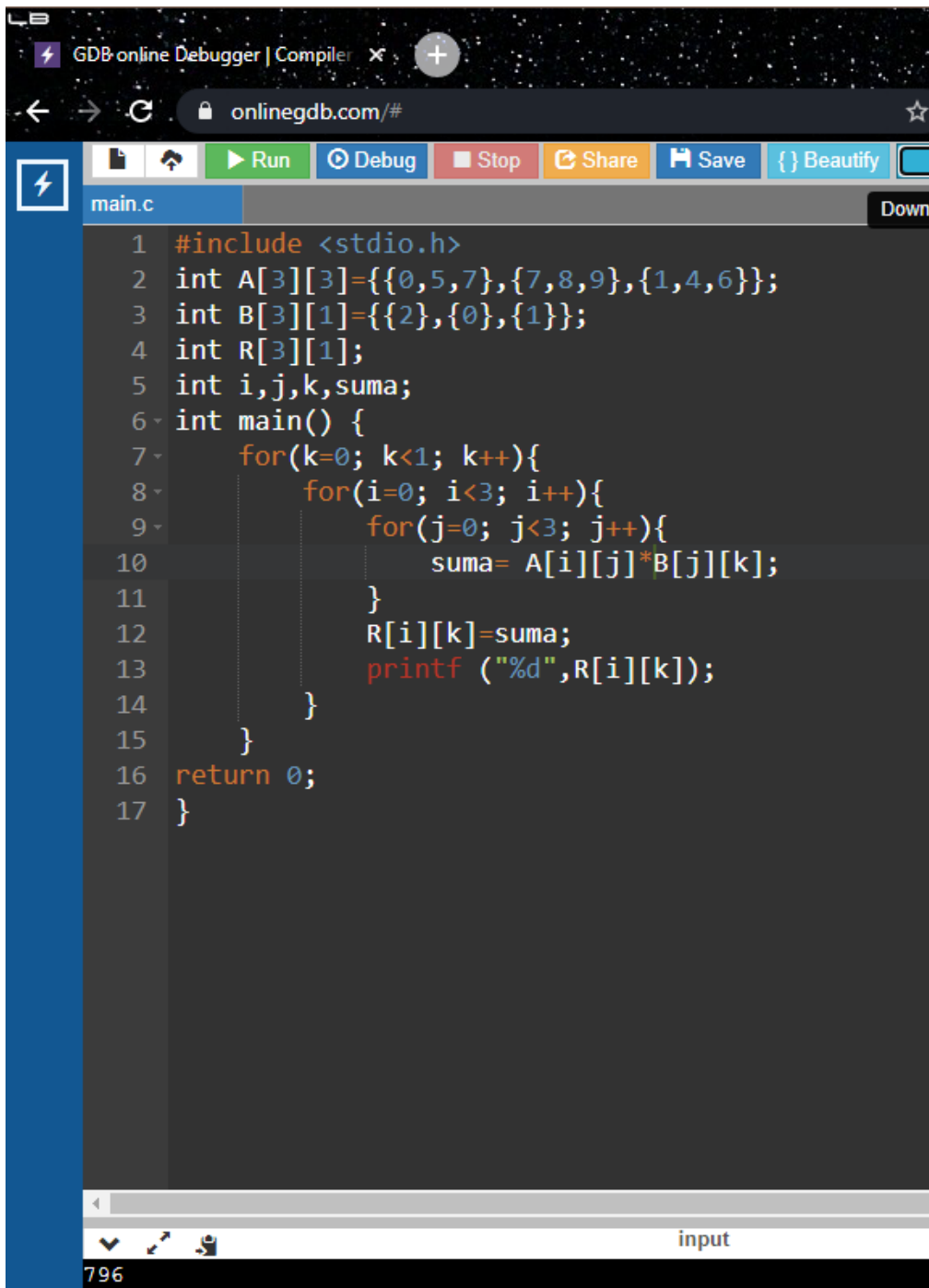
The screenshot shows the GDB online Debugger interface. The top bar includes a lightning bolt icon, the text 'GDB online Debugger | Compiler', and a close button. Below this is a navigation bar with icons for back, forward, refresh, and a URL bar showing 'onlinegdb.com/#'. A toolbar contains buttons for 'Run' (green), 'Debug' (blue), 'Stop' (red), 'Share' (orange), 'Save' (blue), 'Beautify' (light blue), and a download icon. The main editor displays a C program named 'main.c' with the following code:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4 int main ()
5 {
6     int x,y, A[4][4];
7     int f,c, B[4][4], suma, R[4][2],j;
8     srand(time(NULL));
9     printf("Tu matriz de numeros aleatorios es: \n");
10    for(x=0; x<4; x++){
11        for(y=0; y<4; y++){
12            A[x][y]= rand()%5;
13            printf("%d", A[x][y]);
14        }
15        printf ("\n");
16    }
17    printf ("Tu 2da matriz de numeros aleatorios es: \n");
18    for(f=0; f<4; f++){
19        for (c=0; c<4; c++){
20            A[f][c]= rand()%5;
21            printf("%d", A[f][c]);
22        }
23        printf ("\n");
24    }
25    for(j=0; j<4; j++){
26        suma=A[x][y]+B[f][c];
27        R[x][j]= suma;
28    }
29    printf("La suma de tus matrices es: %d\n", R[x][y]);
30 }
31
```

The output window at the bottom shows the following text:

```
1200
Tu 2da matriz de numeros aleatorios es:
5150
5444
1055
0045
La suma de tus matrices es: 5
```


4.8.- Crear una función que multiplique los dos arreglos matriciales e imprima el resultado. Deberá mandar un mensaje de error en caso de que no puedan multiplicarse las matrices.



The screenshot shows the GDB-online Debugger interface. The browser address bar displays 'onlinegdb.com/#'. The interface includes a toolbar with buttons for 'Run', 'Debug', 'Stop', 'Share', 'Save', and 'Beautify'. The file 'main.c' is open, and the code is as follows:

```
1  #include <stdio.h>
2  int A[3][3]={{0,5,7},{7,8,9},{1,4,6}};
3  int B[3][1]={{2},{0},{1}};
4  int R[3][1];
5  int i,j,k,suma;
6  int main() {
7      for(k=0; k<1; k++){
8          for(i=0; i<3; i++){
9              for(j=0; j<3; j++){
10                 suma= A[i][j]*B[j][k];
11             }
12             R[i][k]=suma;
13             printf ("%d",R[i][k]);
14         }
15     }
16     return 0;
17 }
```

At the bottom of the interface, there is an 'input' field and a status bar showing the number '796'.