SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO
Instituto Tecnológico de Iztapalapa

Instituto Tecnològico de iztapalapa

Ingeniería en Sistemas Computacionales

Propuesta para el desarrollo del proyecto del titulado:
"libc++" C++ Standard Library”

Presenta:
Cabrera Ramirez Gerardo
Morales Carrillo Gerardo
Santana Gonzalez Jesus Salvador

No. De Control:
171080187
171080120
171080127

Asesor Interno:
Abiel Tomas Parra Hernandez

Resumen general del proyecto:

El mismo concepto de librería estándar C, pero específico para C ++. La librería estándar C ++ es un conjunto de plantillas de clases C + + que proporciona estructuras de datos de programación común y funciones tales como listas, pilas, matrices, algoritmos, iteradores y cualquier otro componente de C ++ que se te ocurra. La librería estándar C ++ Incorpora la librería estándar de C también y está especificado en el estándar C ++.

Dado que debemos de realizar un proyecto del tema que el profesor nos ha asignado, en nuestro caso fue: “libc++" C++ Standard Library”, deberemos investigar acerca de la plataforma donde se programara y diferentes características que pueden ser implementadas (Depende de lo que el proyecto será a futuro).

En este proyecto se hablará sobre las características y funciones de esta librería ya que tiene muchas cosas de las cuales se pueden aprovechar, como el poder implementarla en otros lenguajes de programación con los cuales sean compatibles. En este caso vamos a corroborar la implementación de "libc++" C++ Standard Library con el lenguaje de programación de python, al crear un proyecto en python y tratar de afirmar que en verdad se pueda compilar el código con la librería de c ++, todo esto siendo documentado paso a paso.

FOTO

EMPRESA: Los mosqueteros TELÉFONO: 5567893456 FAX:
DIRECCIÓN:
ALUMNO: Cabrera Ramirez Gerardo, Morales Carrillo Gerardo, Santana Gonzalez Jesus
CARRERA: Ingeniería en sistemas computacionales
NOMBRE DEL PROYECTO: "libc++" C++ Standard Library
ASESOR INTERNO: Hernandez Parra Abiel
ASESOR EXTERNO: Instituto tecnologico de iztapalapa
FECHA DE INICIO: 26/11/2020 FECHA DE TERMINACIÓN: 24/01/2021
OBJETIVO DEL PROYECTO: Creacion e investigacion de un proyecto relacionado con las librerias C++

Actividad		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Recolección de información	P																
Selección de información que se pondrá en el documento	P																
Buscar programa a realizar	P																

Seleccionar el programa que se realizará	P															
Implementación del programa	P															
Comprobación del programa	P															
Documentación de la implementación	P															
Entrega	P															

Riesgos	Solución
Que un integrante se ausente durante la realización del proyecto (Por enfermedades o sucesos más allá de nuestro control)	Restruturar la forma de trabajo del equipo para cubrir los pendientes que tenga el miembro ausente.
Que demos acceso sin darnos cuenta a cualquiera	Restricciones de acceso solo para usuarios del equipo o que estén trabajando en el proyecto
No tener un respaldo en caso de alguna emergencia (Borrado de datos por ejemplo).	Hacer siempre backups
No entregar el proyecto a tiempo y de manera correcta (Con los requerimientos que se piden).	Hacer un organigrama a seguir para organizar las actividades y tener estructura en los tiempos para no tener ningún percance
Reclamo de algún usuario por derechos de autor por uso de su código.	Contactar al usuario para pedir permiso de utilizar su código.

Descripción general

libc ++ es una nueva implementación de la biblioteca estándar C ++ para C ++ 11 y superior.

Funciones y objetivos

Corrección de acuerdo con la definición del estándar C ++ 11.

Ejecución rápida.

Uso mínimo de memoria.

Tiempo de compilación rápido.

Pruebas unitarias extensivas.

Diseño e implementación.

Pruebas unitarias extensas

El modelo de enlazador interno se puede volcar / leer en formato de texto Se pueden conectar funciones de enlace adicionales "a través de" CPU de factor específico y código específico del sistema operativo

Según la página clang libc ++. (s/e) dice:
A partir de años de experiencia (incluida la implementación de la biblioteca estándar antes), y cambios fundamentales en cómo se implementan. Por ejemplo, generalmente se acepta que construir std :: string usando la “optimización de cadena corta” en lugar de usar Copy On Write (COW) es un enfoque superior para máquinas multinúcleo (particularmente en C ++ 11, que tiene referencias rvalue).Se determinó que romper la compatibilidad de ABI con versiones antiguas de la biblioteca era fundamental para lograr los objetivos de rendimiento de libc ++. Recuperado de. https://libcxx.llvm.org/

La línea principal libstdc ++ cambio a GPL3, que es una licencia que los desarrolladores de libc ++ no pueden usar. libstdc ++ 4.2 (la última versión de GPL2) se puede extender de forma independiente para admitir C ++ 11, pero esta será una rama del código base (para proyectos, generalmente se considera peor que comenzar un nuevo código de forma independiente). Otro problema con la librería es que está integrado con el desarrollo de G ++ y, a menudo, está relacionado con la versión correspondiente de

Justificación:

La realización de este proyecto tiene como finalidad lo siguiente: “Utilizando el intérprete de Python en Visual Studio comprobaremos que se pueda implementar en C++ y que en realidad se pueden hacer implementaciones de python y c++”
Hemos escogido la **metodología tradicional** porque es la que más se acopla ya que nuestro proyecto es de investigación y al final se implementarán las técnicas encontradas para pasar a la etapa de prueba y error

Problema escogido:

Creación de una extensión de C++ para Python

Marco teórico:

Además de los compiladores de C y C ++, también se incluyen ciertos archivos, que generalmente se denominan bibliotecas. La biblioteca contiene el código objeto de muchos programas que le permiten realizar operaciones comunes como leer el teclado, escribir en la pantalla, procesar números, realizar funciones matemáticas, etc.
Aquí hemos de desarrollar el marco teórico de nuestro proyecto, citando ciertos escritos que nos parecieron interesantes durante nuestra investigación:
Según la página https://bcain-llvm.readthedocs.io/ (Sin especificar):

- “La línea principal libstdc ++ se ha cambiado a GPL3, que es una licencia que los desarrolladores de libc ++ no pueden usar. libstdc ++ 4.2 (la última versión de GPL2) se puede extender de forma independiente para admitir C ++ 11, pero esta será una rama del código base (para proyectos, esto generalmente se considera peor que comenzar un nuevo código de forma

independiente). Otro problema con libstdc ++ es que está estrechamente integrado con el desarrollo de G ++ y, a menudo, está estrechamente relacionado con la versión correspondiente de G ++."

- Citando también que: "Las bibliotecas STLport y Apache libstdcxx son otros dos candidatos populares, pero ambas carecen de compatibilidad con C ++ 11. Nuestra experiencia (y la experiencia de los desarrolladores de libstdc ++) es que la adición de soporte para C ++ 11 (especialmente, referencias de rvalue y tipos de solo movimiento) casi requiere cambios en cada clase y función, que en realidad es Volver a escribir. Frente a la reescritura, decidimos partir de cero y evaluar cada decisión de diseño en base a la experiencia basada en primeros principios. "

Recuperado de https://docs.google.com/document/d/1crku6_1vpmVRZIKbo00sk5JMsmKQuROOj7EMpCCgie4/edit, 03/12/2020

El front-end de C ++ de GCC 11 (G ++) probablemente ofrecerá soporte para usar la biblioteca estándar libc ++ de LLVM. Recientemente, se hizo una pregunta en la lista de correo de GCC sobre la capacidad de hacer *-stdlib = libc* ++ para usar la biblioteca estándar C ++ de LLVM junto con el compilador GCC C ++.

Aquellos que se pregunten acerca de las capacidades actuales de libc ++ de LLVM pueden encontrarlo a través de libcxx.llvm.org.

Desarrollo:

La página de microsoft nos dice que los requerimientos para realizar la comprobación son:

- Visual Studio 2017 o versiones posteriores con las cargas de trabajo Desarrollo para el escritorio con C++ y Desarrollo de Python instaladas con opciones predeterminadas.
- En la carga de trabajo Desarrollo de Python, seleccione también el cuadro de la derecha de Herramientas de desarrollo nativo de Python. Esta opción establece la mayor parte de la configuración descrita en este artículo. (Esta opción también incluye la carga de trabajo de C++ automáticamente).

1.- Haremos la creación e implementación de c++ para phyton, revisaremos los requisitos que se necesiten:

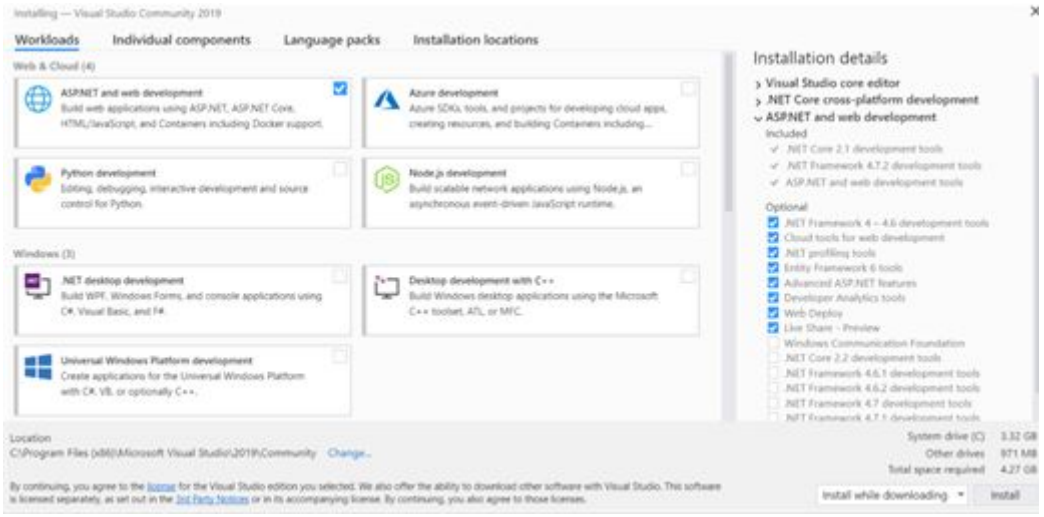
*Visual studio 2017 (o posteriores) con las cargas de trabajo (Desarrollo para el escritorio c++ y desarrollo de phyton)

*Primero instalaremos visual studio de primera instancia, el cual será obtenido de la página de Microsoft

Paso 1:

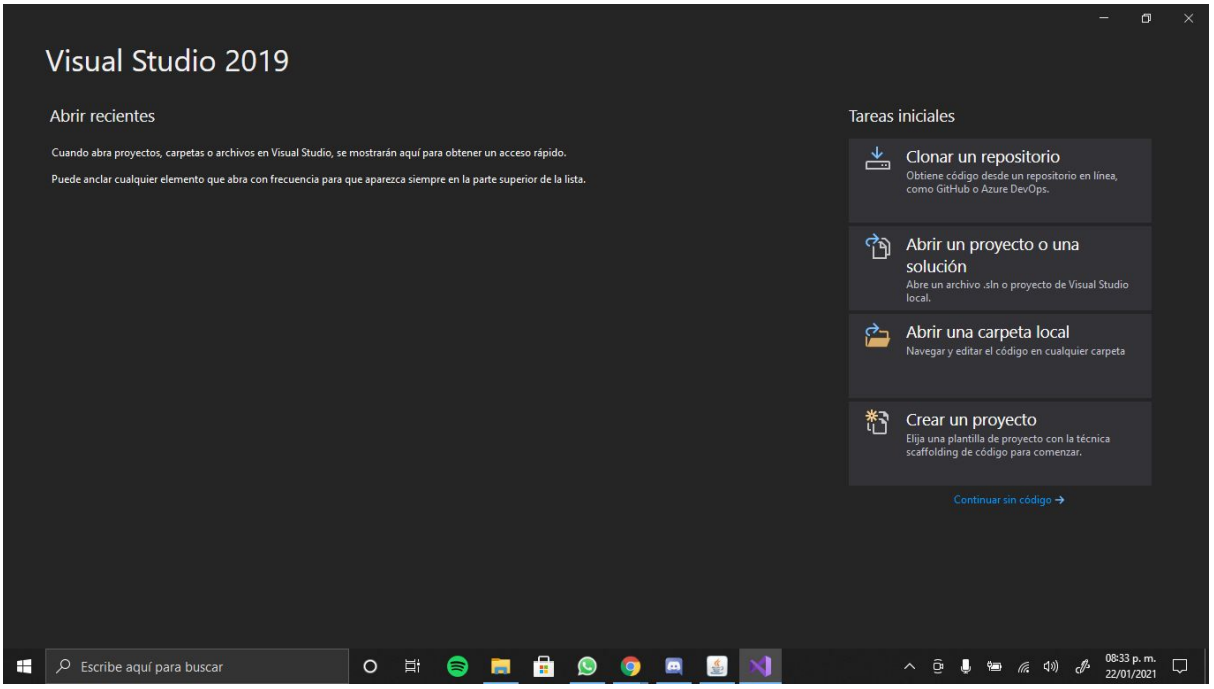
- Comprobar los requisitos del sistema para comprobar compatibilidad con el programa.
- Ya descargado visual studio, haremos doble click en el archivo .exe
- A continuación, veremos una pantalla en la cual daremos en la opción “si” para ejecutar el programa, términos y condiciones: aceptar, aparecerá una nueva ventana en donde elegiremos las cargas de trabajo que necesitamos, en este caso, seran las siguientes:

· Desarrollo para el escritorio c++ y desarrollo de phyton

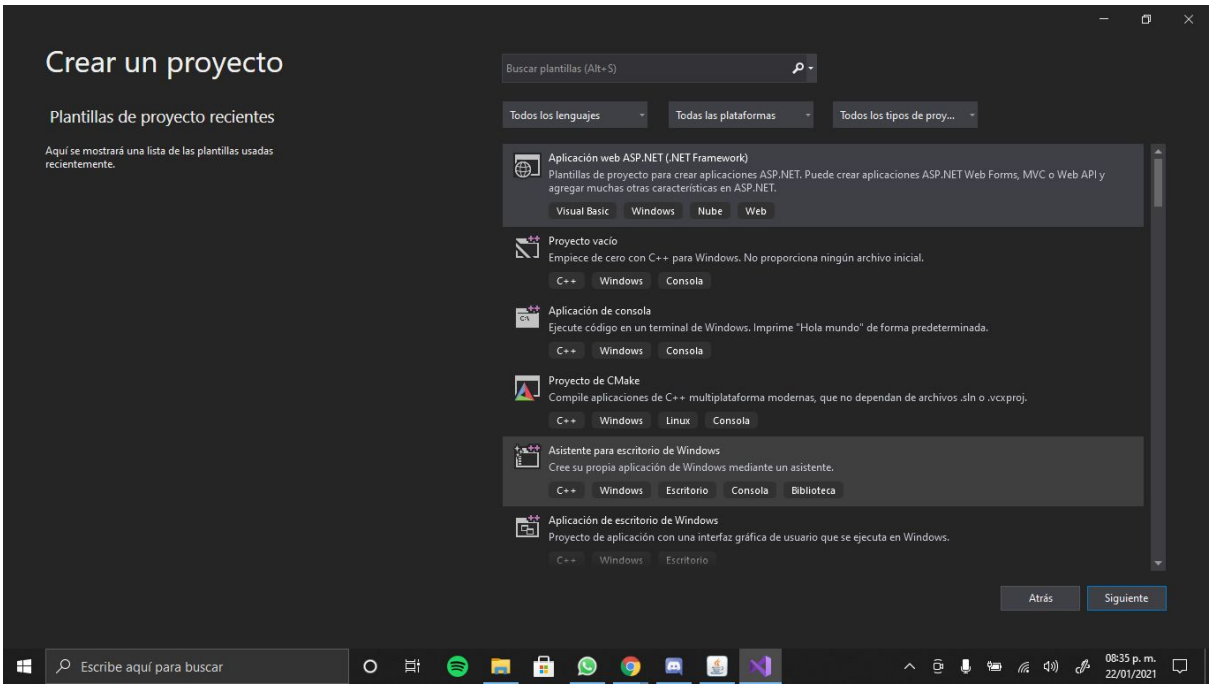


Esto es para obtener la compatibilidad con c y c++, cuando la instalación de visual studio se complete, daremos en el botón “Iniciar”.

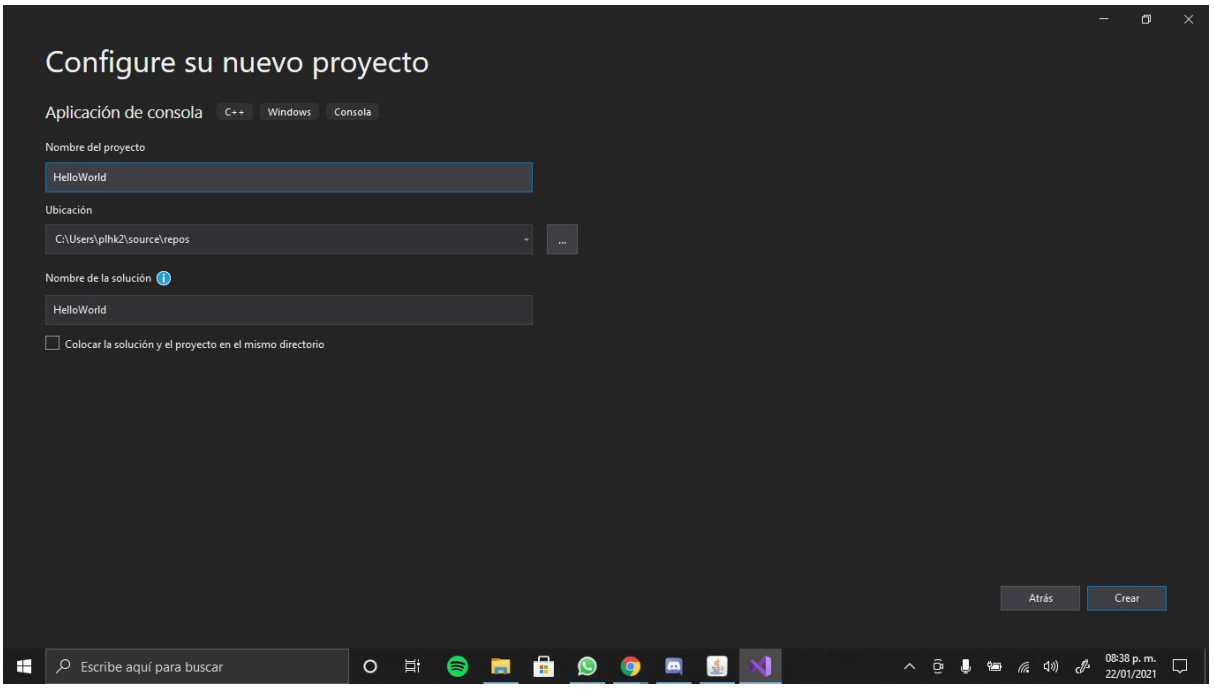
Estaremos en la ventana de inicio, elegiremos “Crear un proyecto nuevo”



Accederemos al cuadro de búsqueda, donde elegiremos “Proyecto vacío”

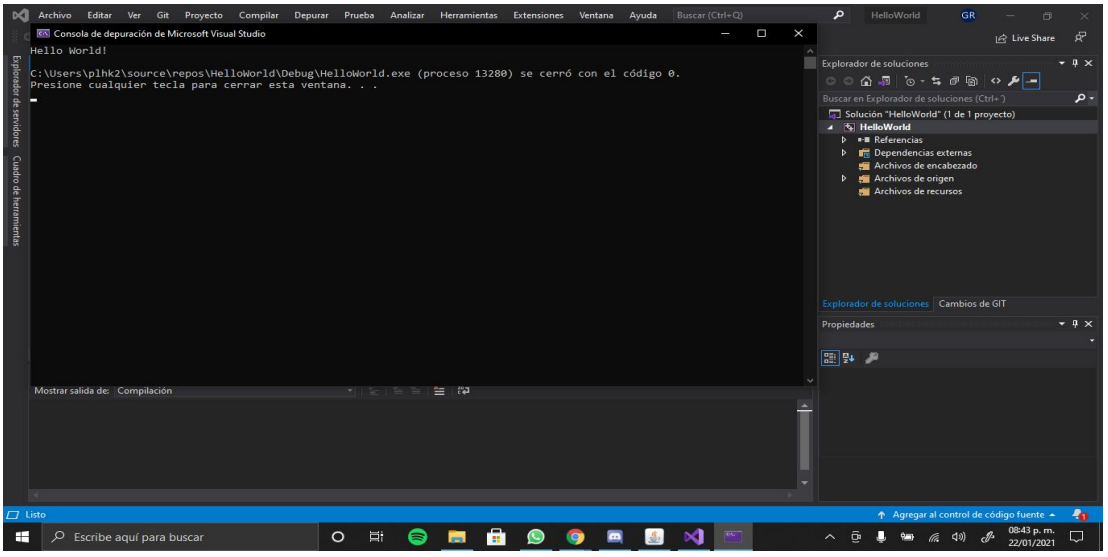


Daremos en siguiente, ahora, seleccionaremos “console app”, se abrirá un cuadro de diálogo , “Configurar el nuevo proyecto”, haremos el conocido “Hello world”

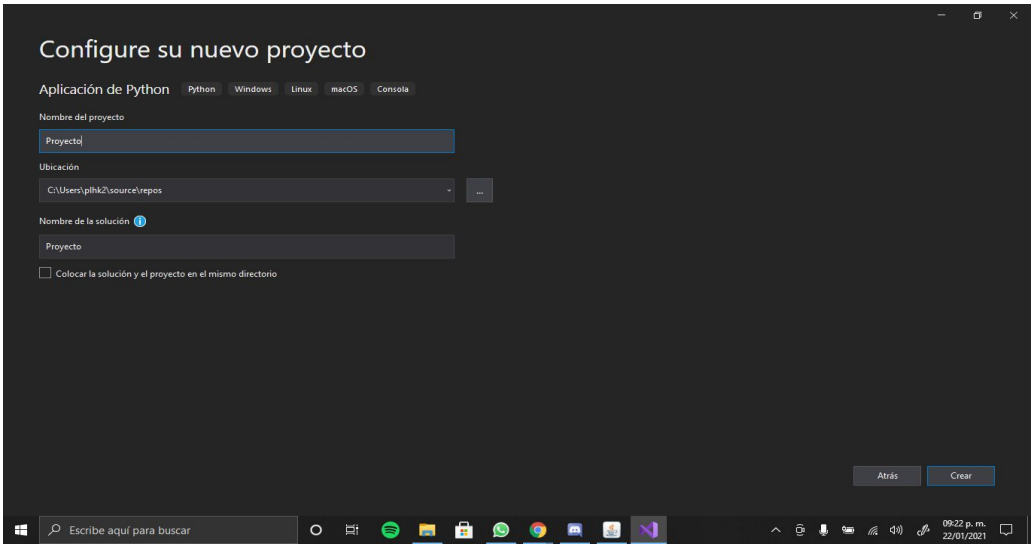


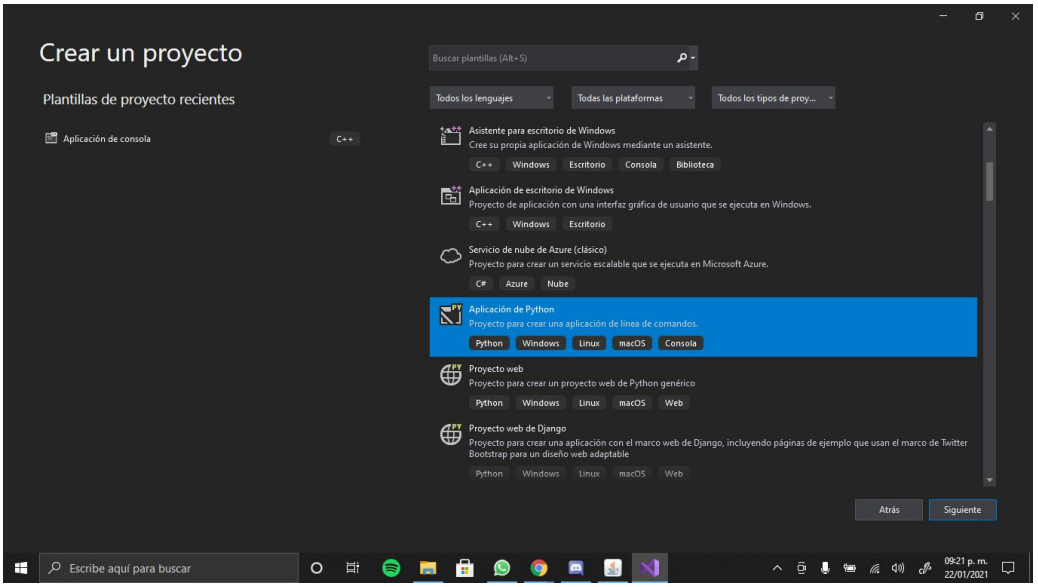
Daremos en “Crear”:

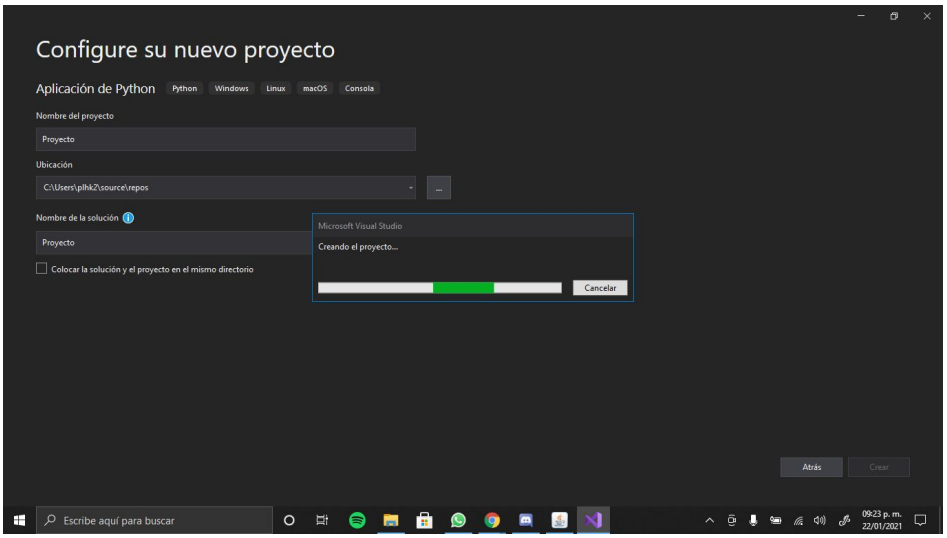
Haremos click en el botón “Compilar”, “Compilar solución”, y para ejecutar el código, le daremos a “Depurar”, “Iniciar sin depurar”, aparecerá una nueva ventana:



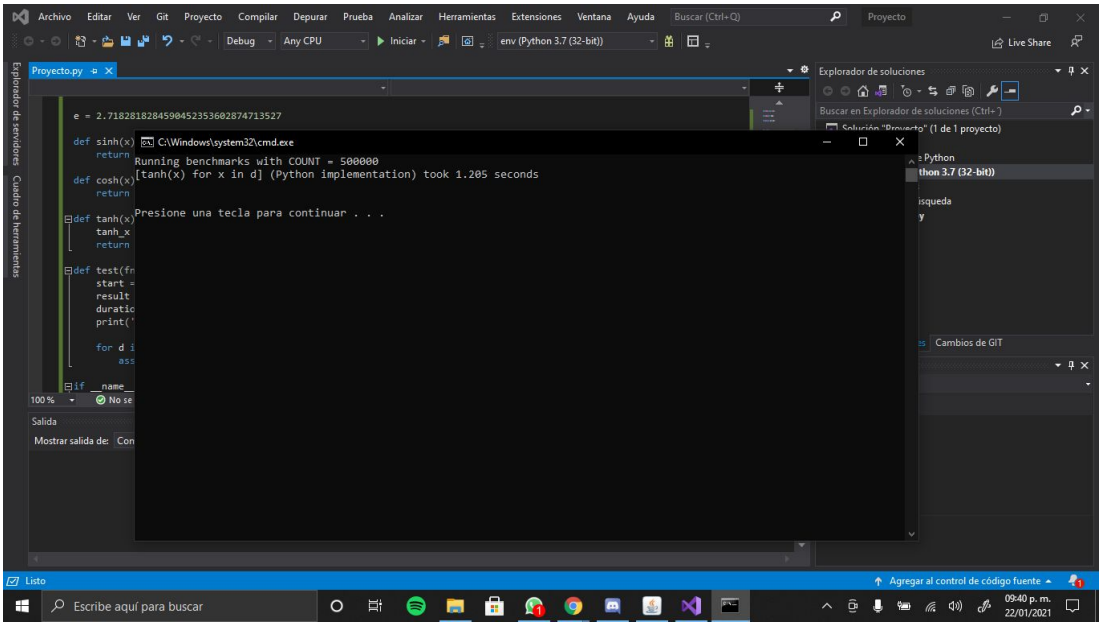
Después de instalar Visual Studio y haber creado un proyecto de C ++, compilarlo y depurarlo, empezaremos con la creación de la extensión de C ++ para Python.
Crearemos un proyecto de Python en Visual Studio, seleccionamos la plantilla de aplicación de Python y le daremos en siguiente, le daremos nombre y en crear





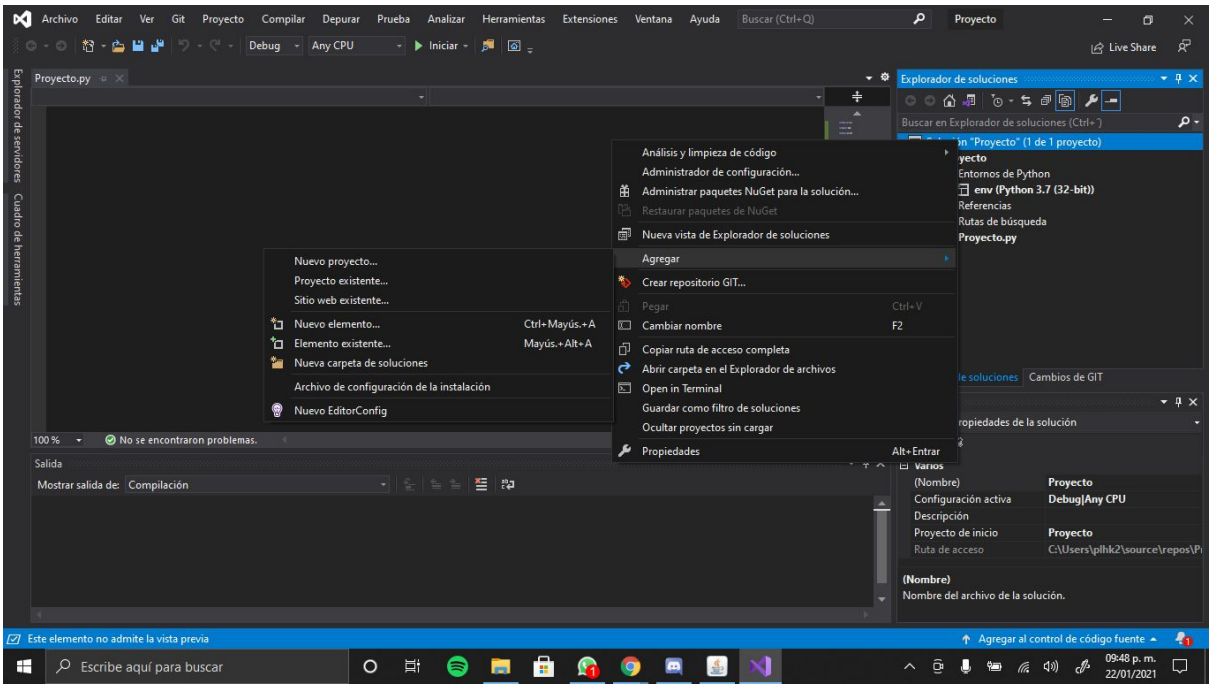


En la página nos recomiendan usar el intérprete de Python de 32 bits. En el archivo .py pegaremos el código que nos proporciona la página el cual es una prueba comparativa del cálculo de una tangente hiperbólica, después de eso ejecutaremos el programa mediante depurar y luego iniciar sin depurar.

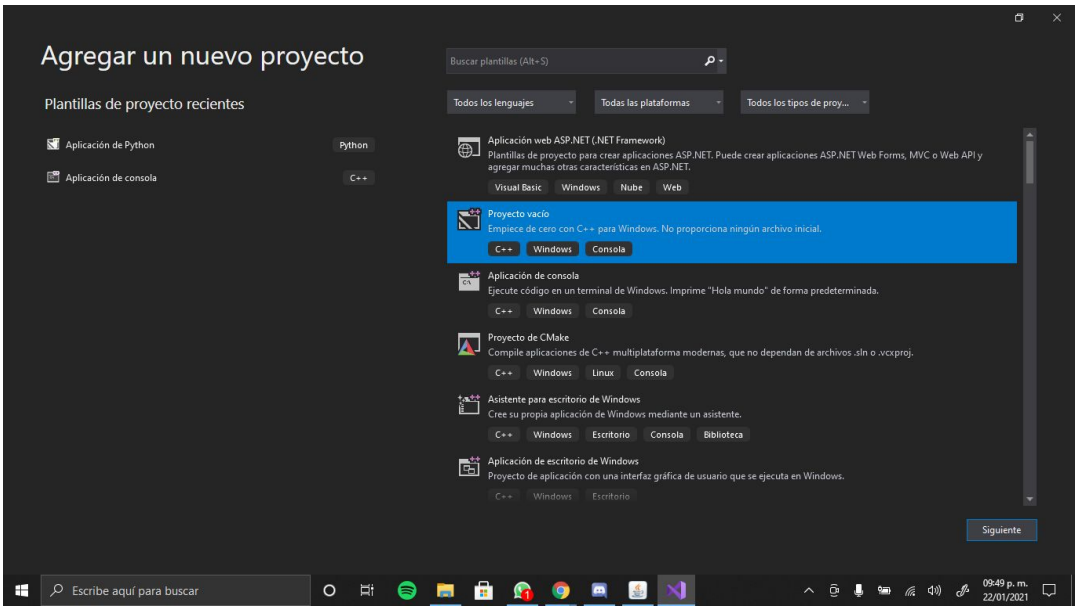


Seguiremos las instrucciones que se nos indican para crear dos proyectos de C ++ idénticos denominados “superfastcode” y “superfastcode2”. Despues de

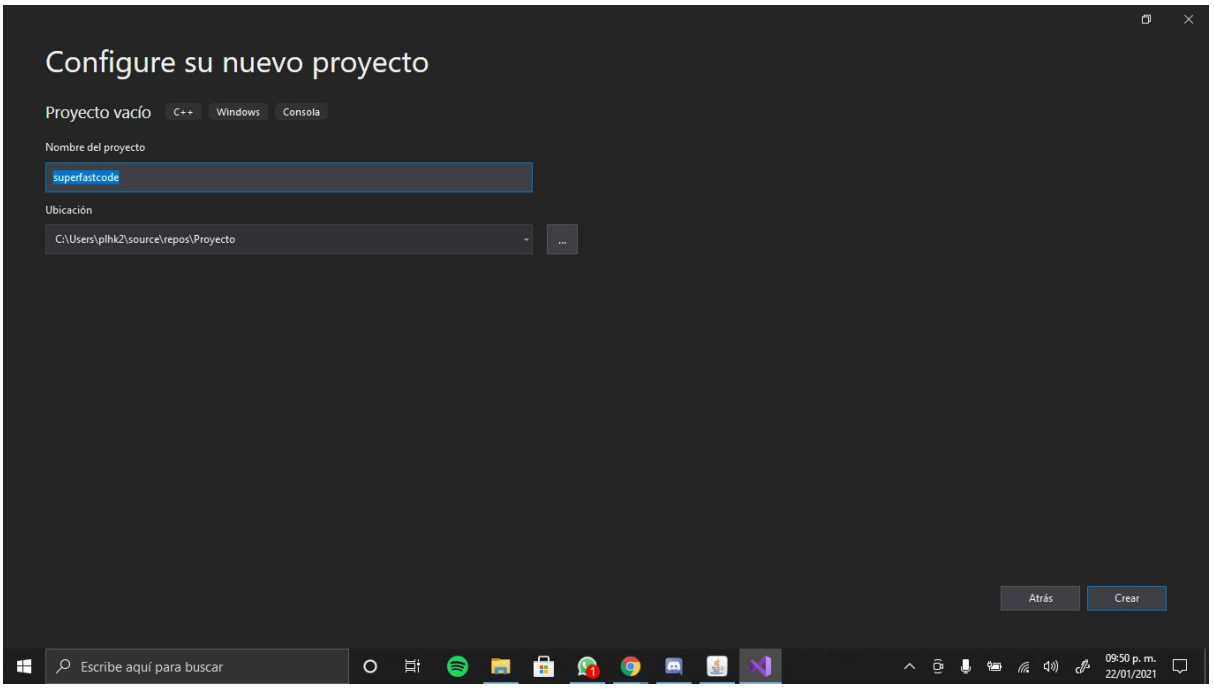
eso vamos a dar clic derecho en el explorador de soluciones y seleccione agregar



Y luego en nuevo proyecto, buscaremos C ++ y seleccionaremos proyecto vacío

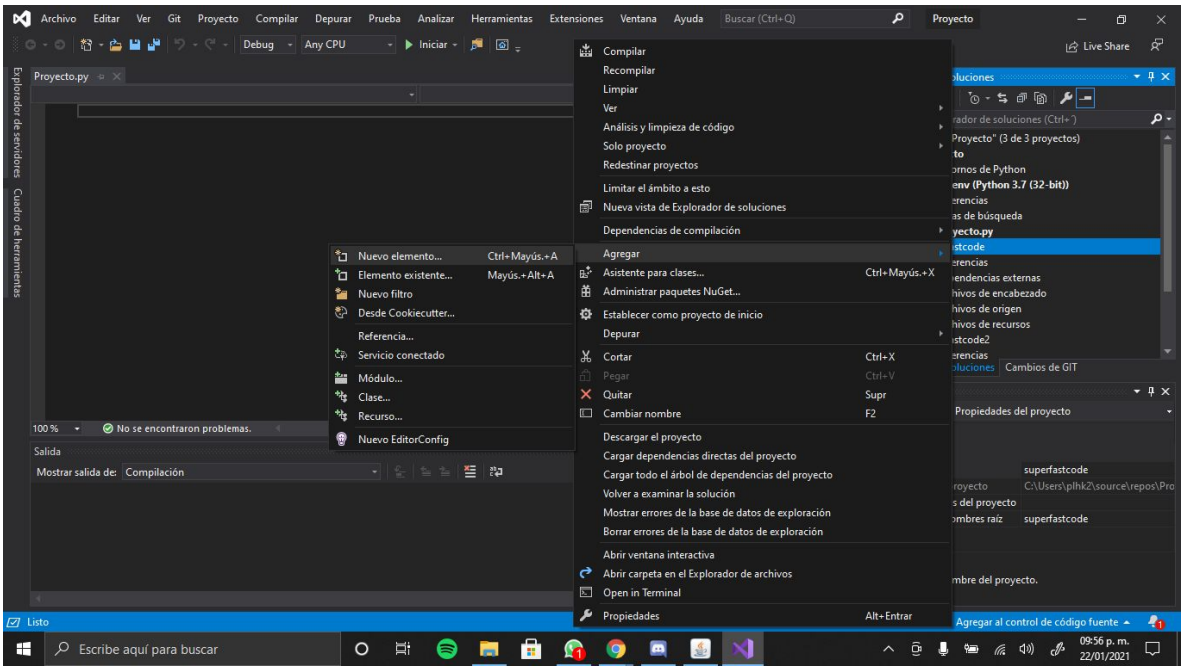


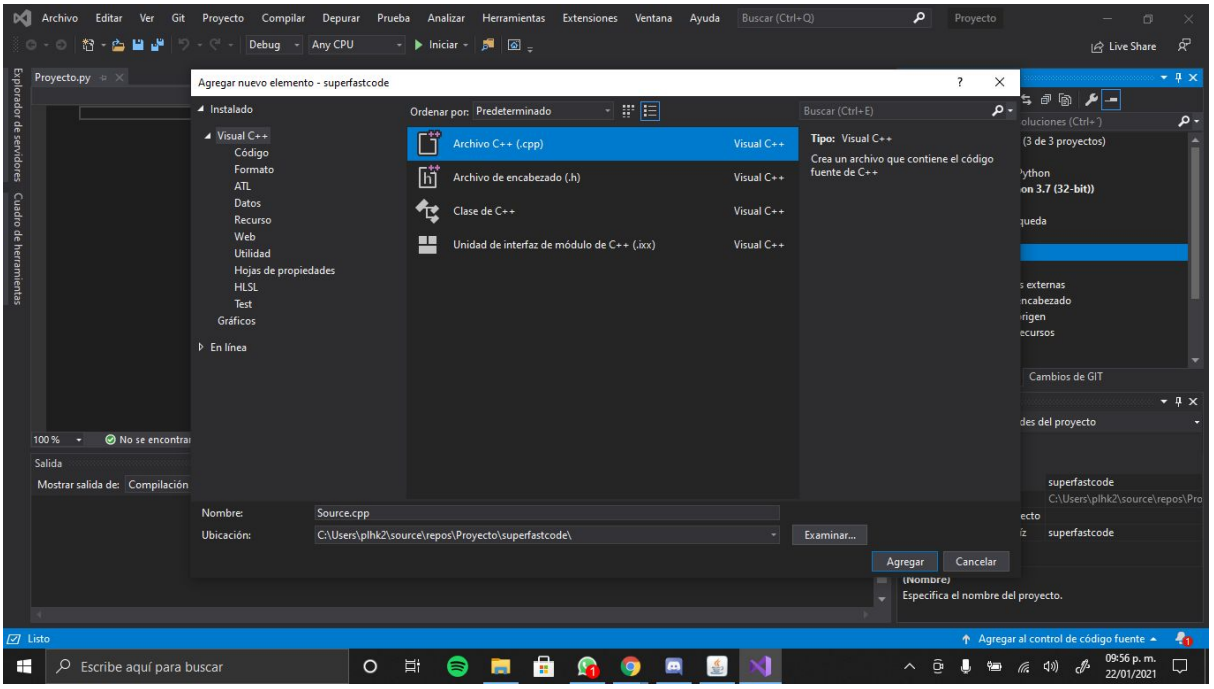
Y le daremos el nombre de superfastcode



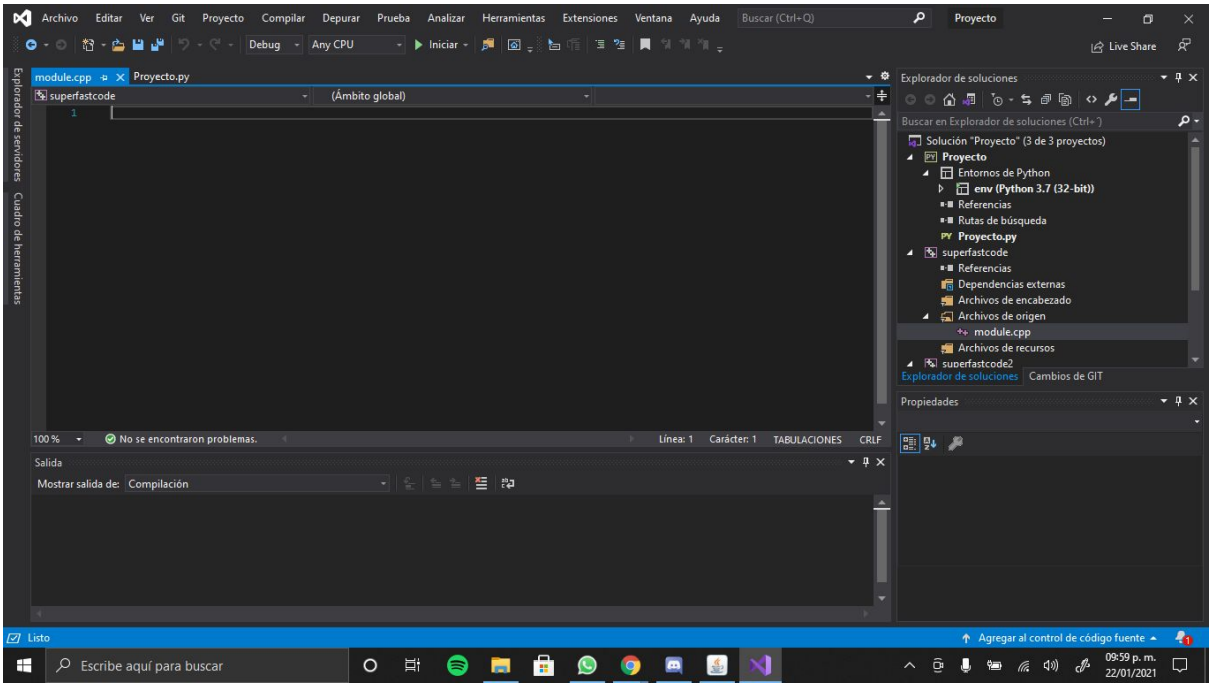
Y haremos otro proyecto con el nombre superfastcode2.

Crearemos un archivo de C ++, para ello hacemos clic derecho en el archivo del código fuente y en agregar nuevo elemento y agregamos un archivo de C++ en .cpp

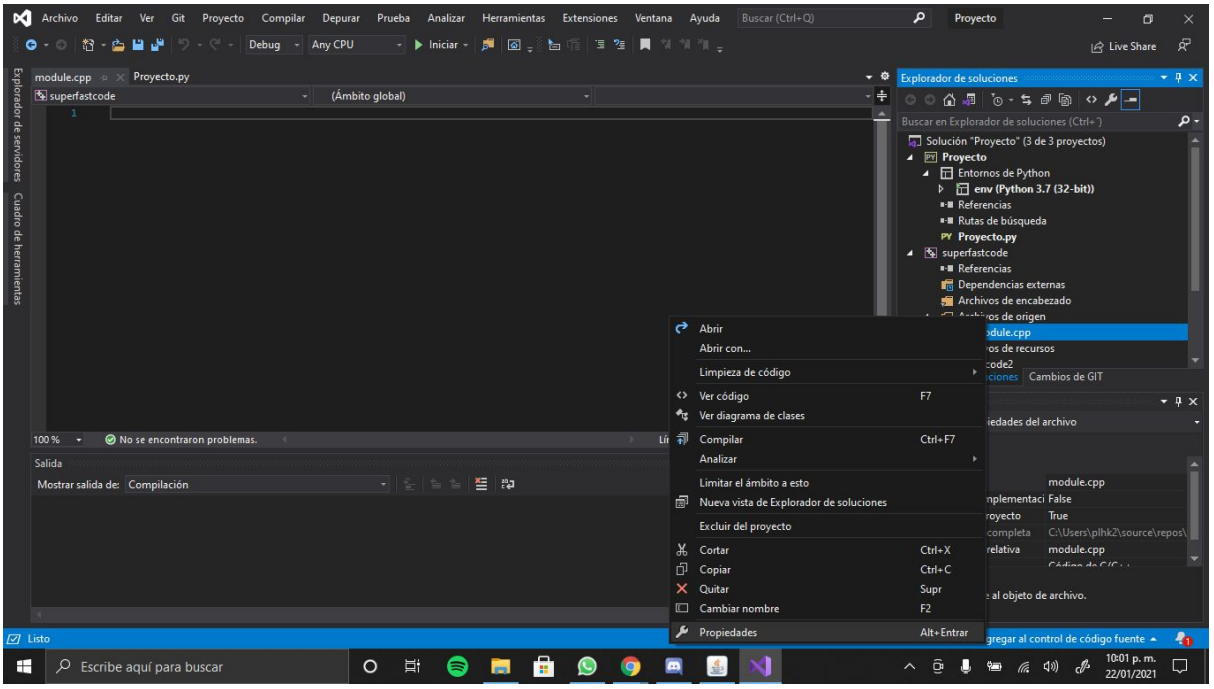

Archivo Editar Ver Git Proyecto Compilar Depurar Prueba Analizar Herramientas Extensiones Ventana Ayuda
Buscar (Ctrl+Q)
Proyecto
Debug
Any CPU
Iniciar
Live Share
Proyecto.py
Compilar
Recompilar
Limpiar
Ver
Análisis y limpieza de código
Solo proyecto
Redestinar proyectos
Limitar el ámbito a esto
Nueva vista de Explorador de soluciones
Dependencias de compilación
Agregar
Asistente para clases... Ctrl+Mayús.+X
Administrar paquetes NuGet...
Establecer como proyecto de inicio
Depurar
Cortar Ctrl+X
Pegar Ctrl+V
Quitar Supr
Cambiar nombre F2
Descargar el proyecto
Cargar dependencias directas del proyecto
Cargar todo el árbol de dependencias del proyecto
Volver a examinar la solución
Mostrar errores de la base de datos de exploración
Borrar errores de la base de datos de exploración
Abrir ventana interactiva
Abrir carpeta en el Explorador de archivos
Open in Terminal
Propiedades Alt+Entrar
Nuevo elemento... Ctrl+Mayús.+A
Elemento existente... Mayús.+Alt+A
Nuevo filtro
Desde Cookiecutter...
Referencia...
Servicio conectado
Módulo...
Clase...
Recurso...
Nuevo EditorConfig
No se encontraron problemas.
Salida
Mostrar salida de: Compilación
Cambios de GIT
Propiedades del proyecto
Listo
Agregar al control de código fuente
Escribe aquí para buscar
09:56 p. m.
22/01/2021


Archivo Editar Ver Git Proyecto Compilar Depurar Prueba Analizar Herramientas Extensiones Ventana Ayuda
Buscar (Ctrl+Q)
Proyecto
Debug
Any CPU
Iniciar
Live Share
Proyecto.py
Agregar nuevo elemento - superfastcode
Instalado
Visual C++
Código
Formato
ATL
Datos
Recurso
Web
Utilidad
Hojas de propiedades
HLSL
Test
Gráficos
En línea
Ordenar por: Predeterminado
Archivo C++ (.cpp) Visual C++
Archivo de encabezado (.h) Visual C++
Clase de C++ Visual C++
Unidad de interfaz de módulo de C++ (.ixx) Visual C++
Tipo: Visual C++
Crea un archivo que contiene el código fuente de C++
Nombre: Source.cpp
Ubicación: C:\Users\plhk2\source\repos\Proyecto\superfastcode\
Examinar...
Agregar
Cancelar
Especifica el nombre del proyecto.
Cambios de GIT
Listo
Agregar al control de código fuente
Escribe aquí para buscar
09:56 p. m.
22/01/2021

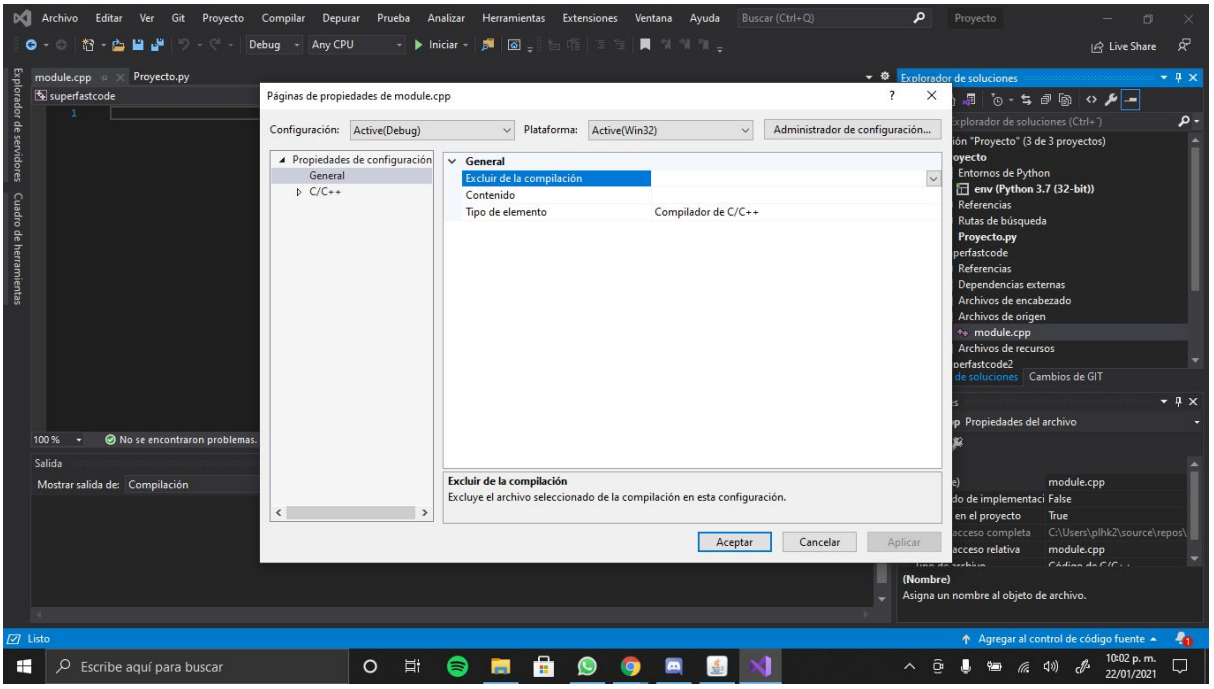
Cambiaremos el nombre por “module.cpp”

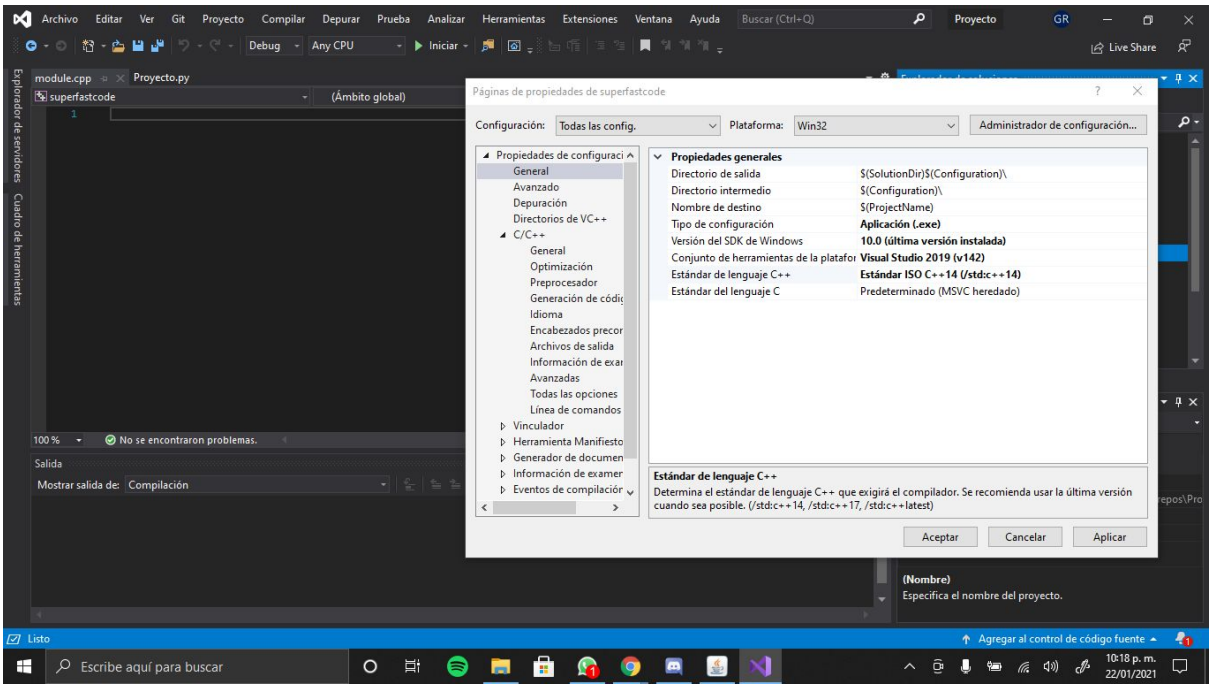


hacemos clic derecho en el proyecto de C++ y le daremos en propiedades



y estableceremos la plataforma como win 32



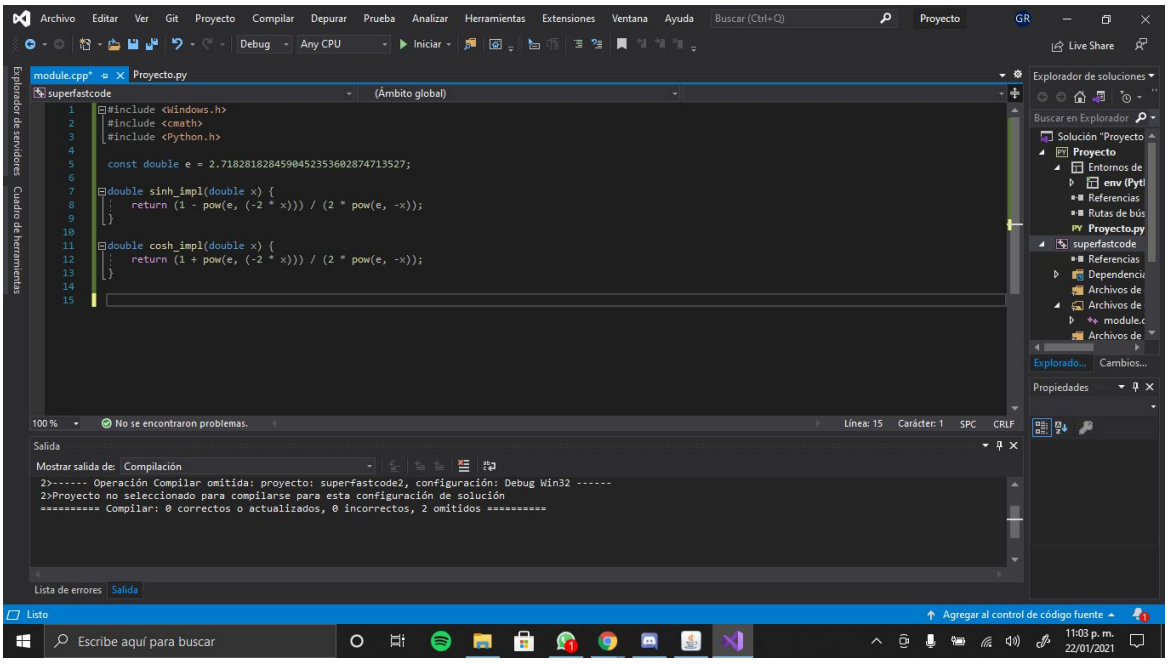


Convertiremos los proyectos de C ++ en extensiones para python

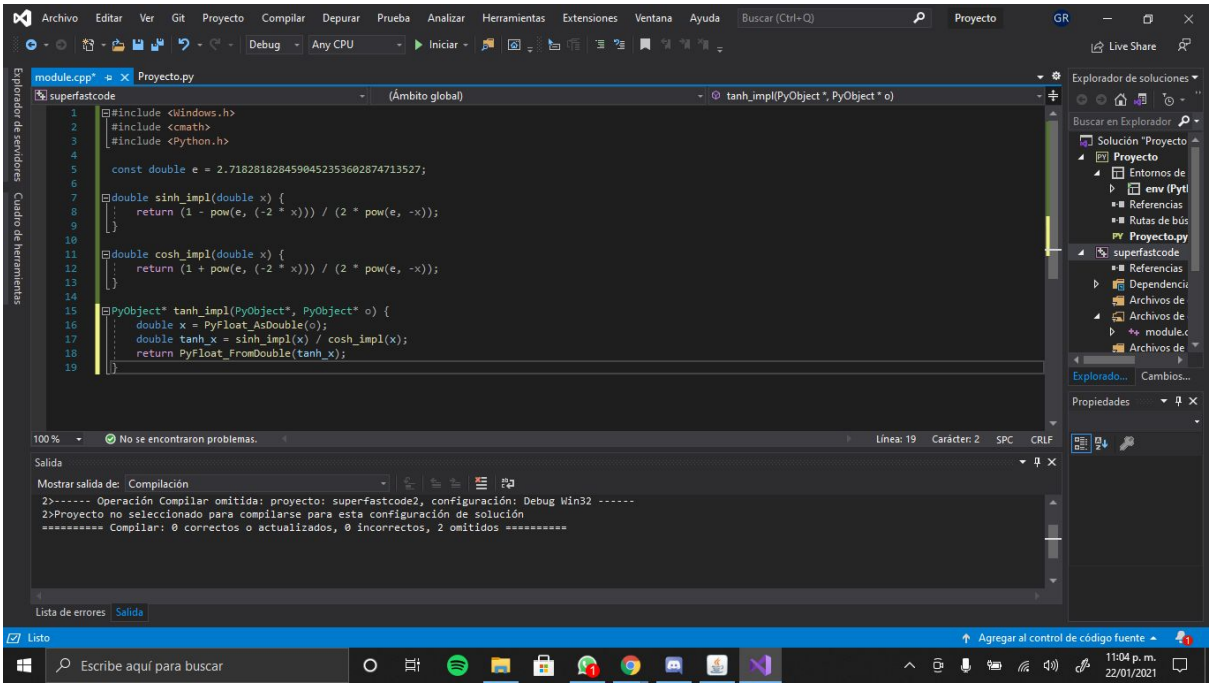
Para convertir el dll en una extensión para python debemos de modificar los métodos exportados de modo que interactúen con tipos de python, para ello debemos agregar una función que exporte el module.

Extensiones de C python

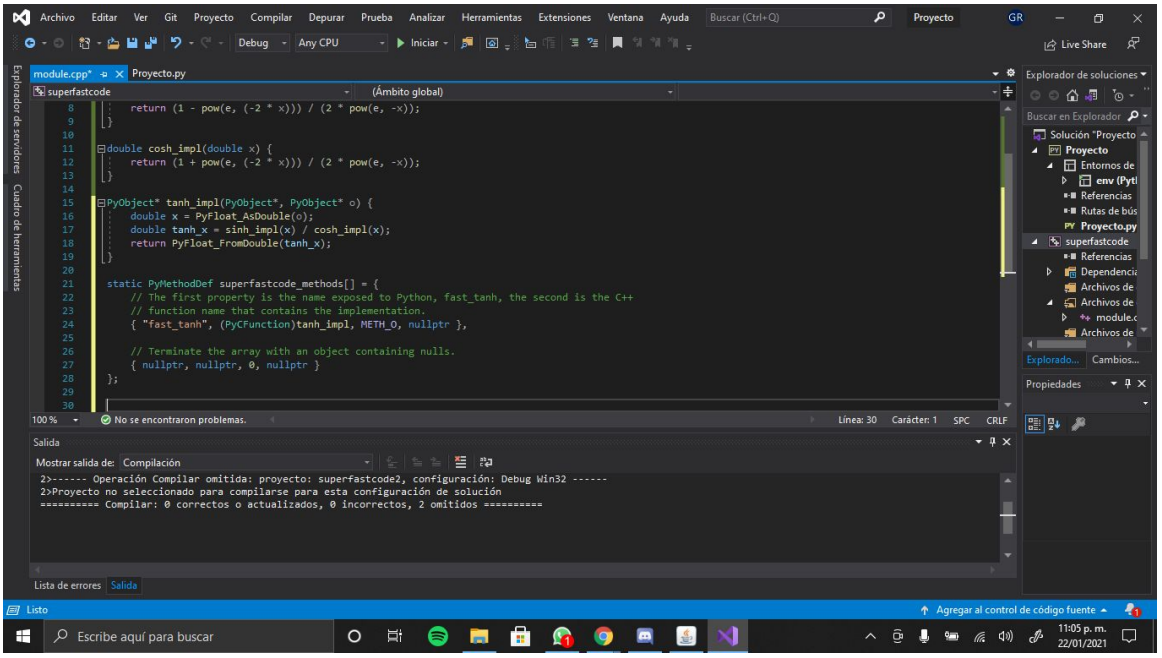
1- En la parte superior del module.cpp incluiremos python.h



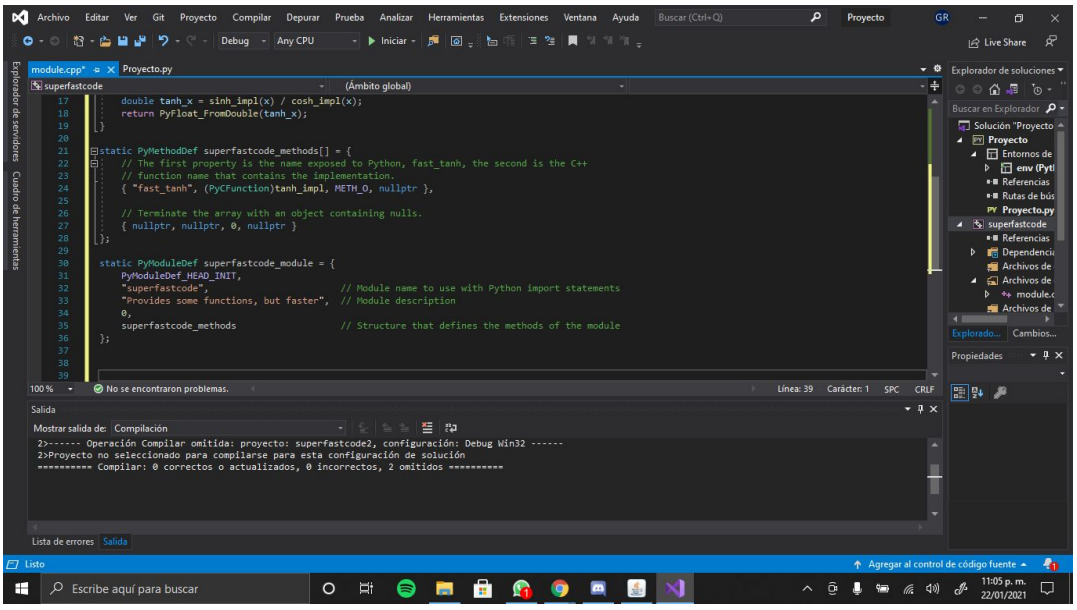
2- Modifica el método tanh_impl para aceptar y devolver tipos de Python (PyObject*).



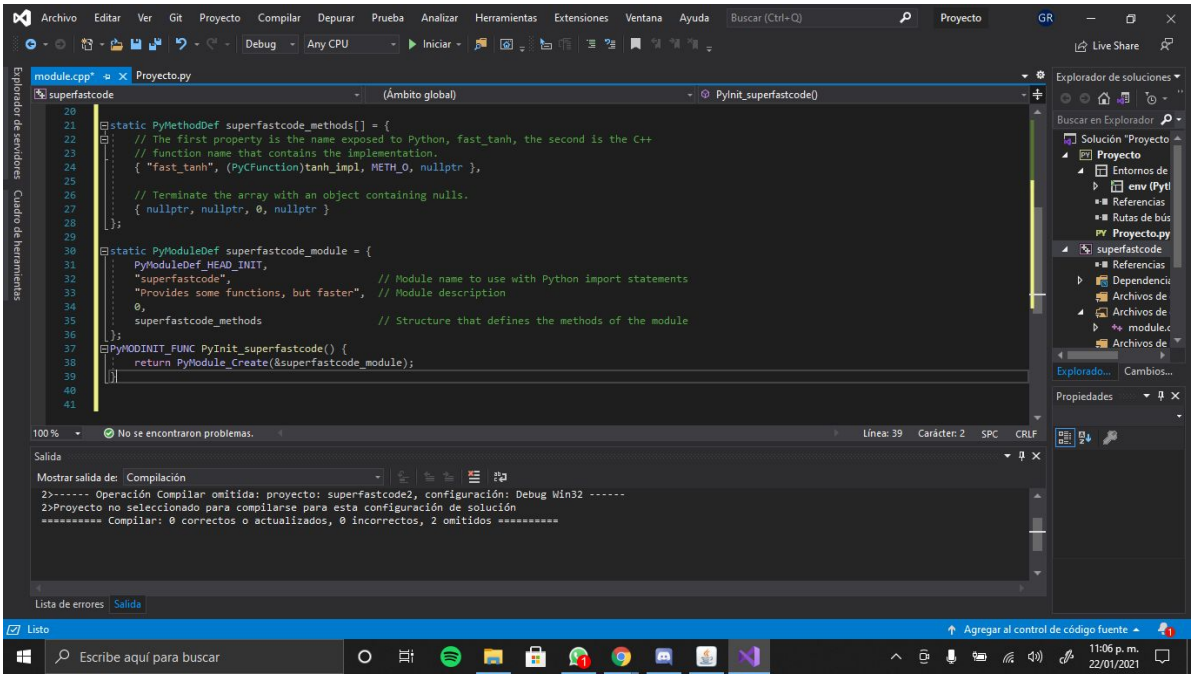
3- Agregue una estructura que defina la manera en que la función tanh_impl de C++ se muestra en Python:



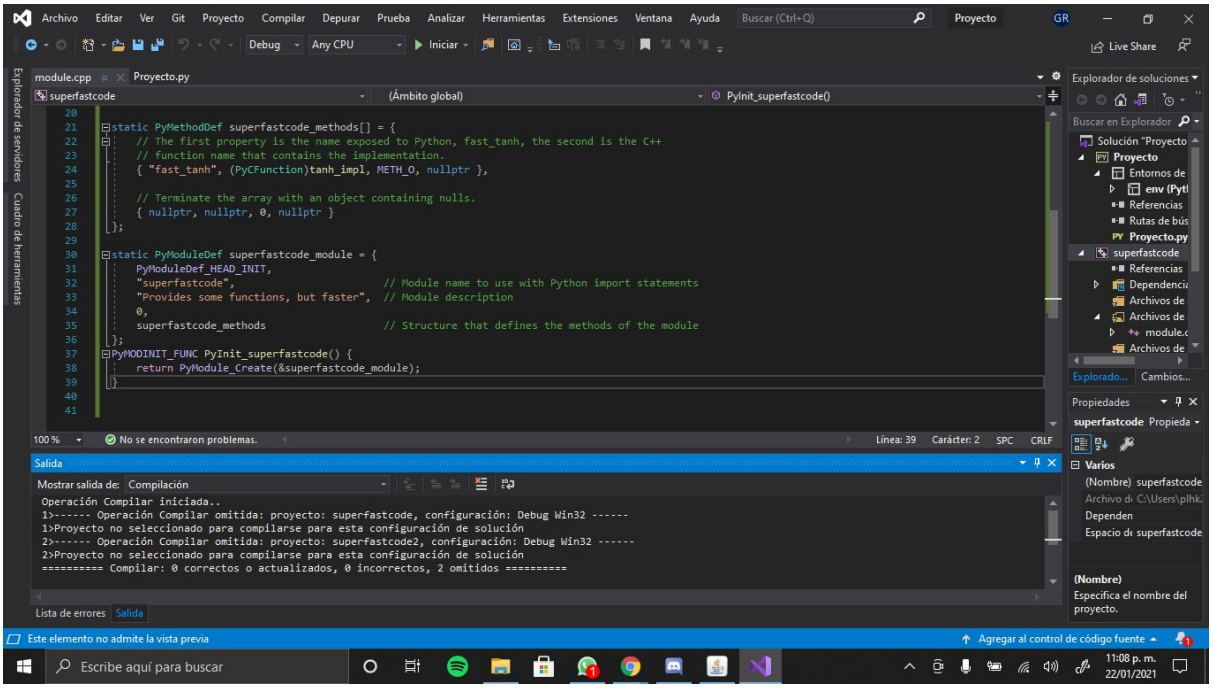
4- Agregue una estructura que defina el módulo como quiere que se haga referencia a él en el código de Python, concretamente cuando se usa la instrucción from...import. (Hágalo coincidir con el valor de las propiedades del proyecto en Propiedades de configuración > General > Nombre de destino). En el ejemplo siguiente, el nombre de módulo "superfastcode" significa que puede usar from superfastcode import fast_tanh en Python, porque fast_tanh se define en superfastcode_methods. (Los nombres de archivo internos del proyecto de C++, como module.cpp, no tienen importancia).



5- Agregue un método al que Python llame cuando cargue el módulo, con el nombre PyInit_<module-name>, donde <module_name> debe coincidir exactamente con la propiedad General > Nombre de destino del proyecto de C++ (es decir, coincide con el nombre del archivo de .pyd compilado por el proyecto).

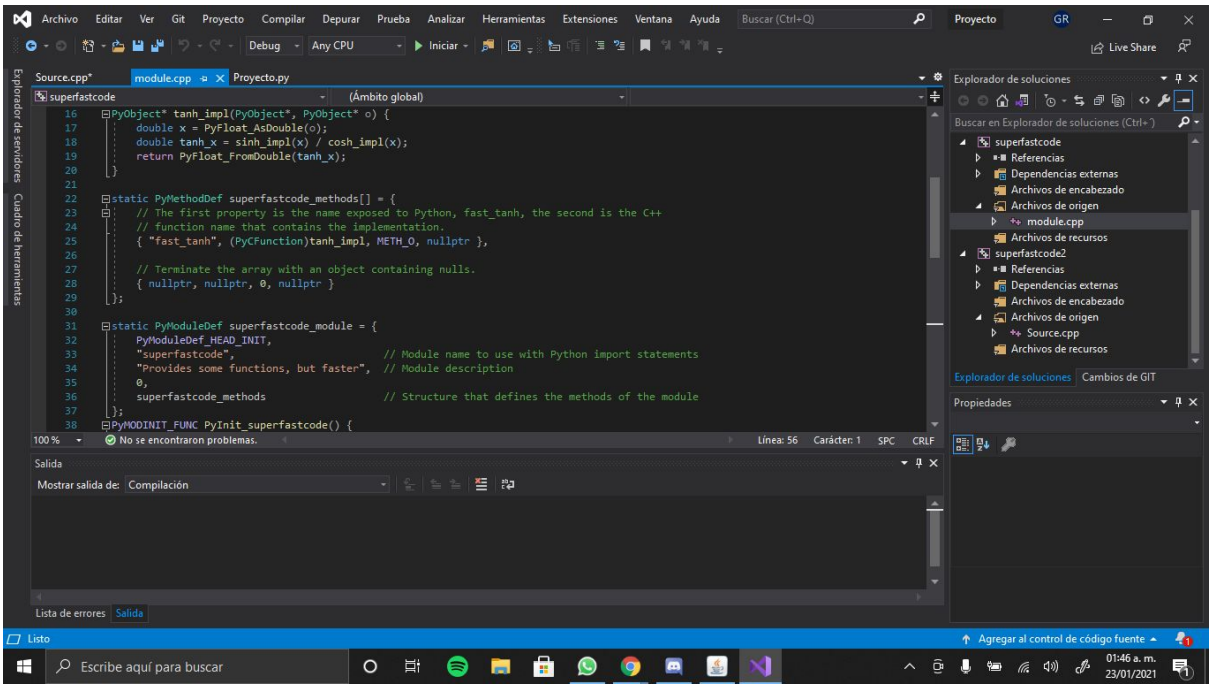


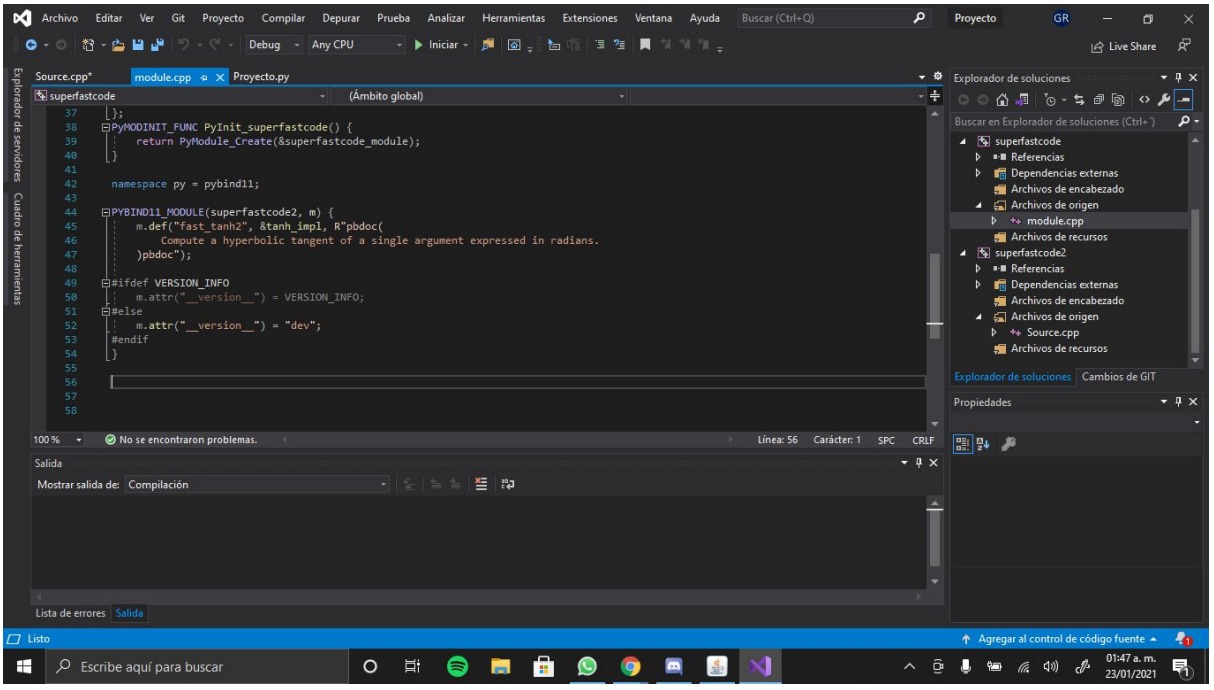
6-Establezca la configuración de destino en Lanzamiento y compile el proyecto de C++ de nuevo para comprobar el código.



Pybin11

1-Instale PyBind11 mediante pip: pip install pybind11 o py -m pip install pybind11
2-En la parte superior de module.cpp, incluya pybind11.h:
3-En la parte inferior de module.cpp, use la macro PYBIND11_MODULE para definir el punto de entrada a la función de C++:
4-Establezca la configuración de destino en Lanzamiento y compile el proyecto de C++ para comprobar el código. Si se producen errores, vea la sección Solución de problemas a continuación.

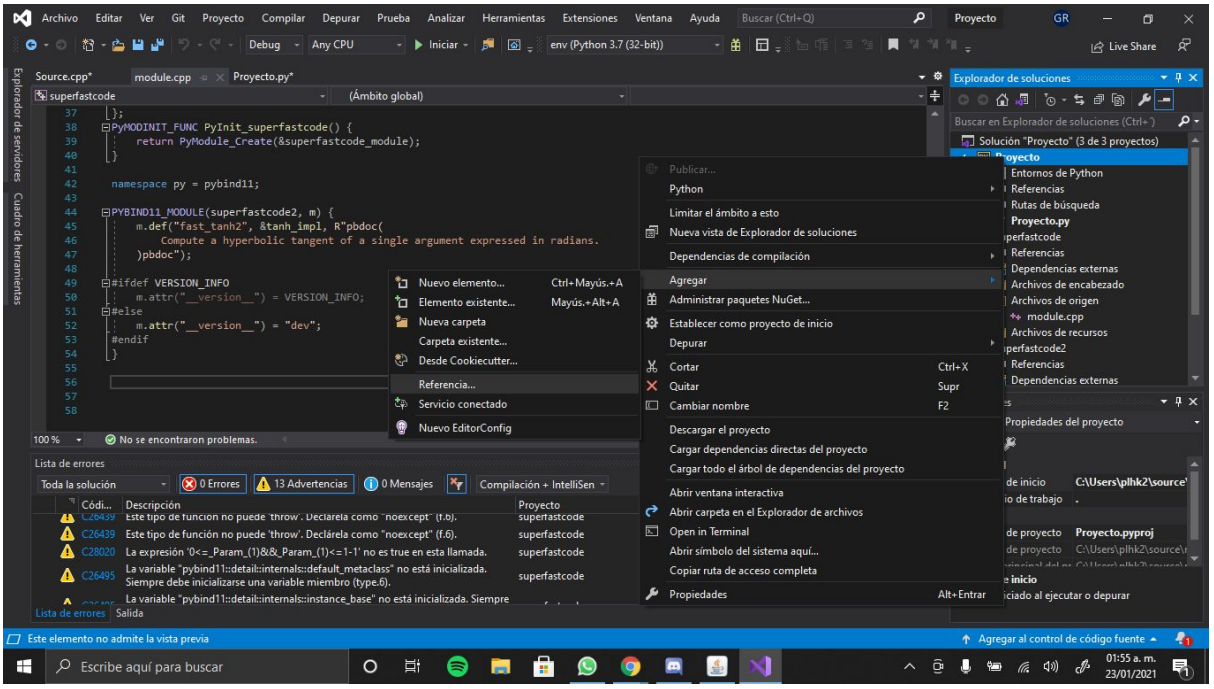

Archivo Editar Ver Git Proyecto Compilar Depurar Prueba Analizar Herramientas Extensiones Ventana Ayuda
Buscar (Ctrl+Q)
Proyecto
Debug
Any CPU
Iniciar
Live Share
Source.cpp*
module.cpp
Proyecto.py
superfastcode
(Ámbito global)
PyObject* tanh_impl(PyObject*, PyObject* o) {
double x = PyFloat_AsDouble(o);
double tanh_x = sinh_impl(x) / cosh_impl(x);
return PyFloat_FromDouble(tanh_x);
}
static PyMethodDef superfastcode_methods[] = {
// The first property is the name exposed to Python, fast_tanh, the second is the C++
// function name that contains the implementation.
{ "fast_tanh", (PyCFunction)tanh_impl, METH_O, nullptr },
// Terminate the array with an object containing nulls.
{ nullptr, nullptr, 0, nullptr }
};
static PyModuleDef superfastcode_module = {
PyModuleDef_HEAD_INIT,
"superfastcode", // Module name to use with Python import statements
"Provides some functions, but faster", // Module description
0,
superfastcode_methods // Structure that defines the methods of the module
};
PyMODINIT_FUNC PyInit_superfastcode() {
100 %
No se encontraron problemas.
Línea: 56 Carácter: 1 SPC CRLF
Salida
Mostrar salida de: Compilación
Lista de errores Salida
Explorador de soluciones
Buscar en Explorador de soluciones (Ctrl+;)
superfastcode
Referencias
Dependencias externas
Archivos de encabezado
Archivos de origen
module.cpp
Archivos de recursos
superfastcode2
Referencias
Dependencias externas
Archivos de encabezado
Archivos de origen
Source.cpp
Archivos de recursos
Explorador de soluciones Cambios de GIT
Propiedades
Listo
Agregar al control de código fuente
Escribe aquí para buscar
01:46 a. m.
23/01/2021

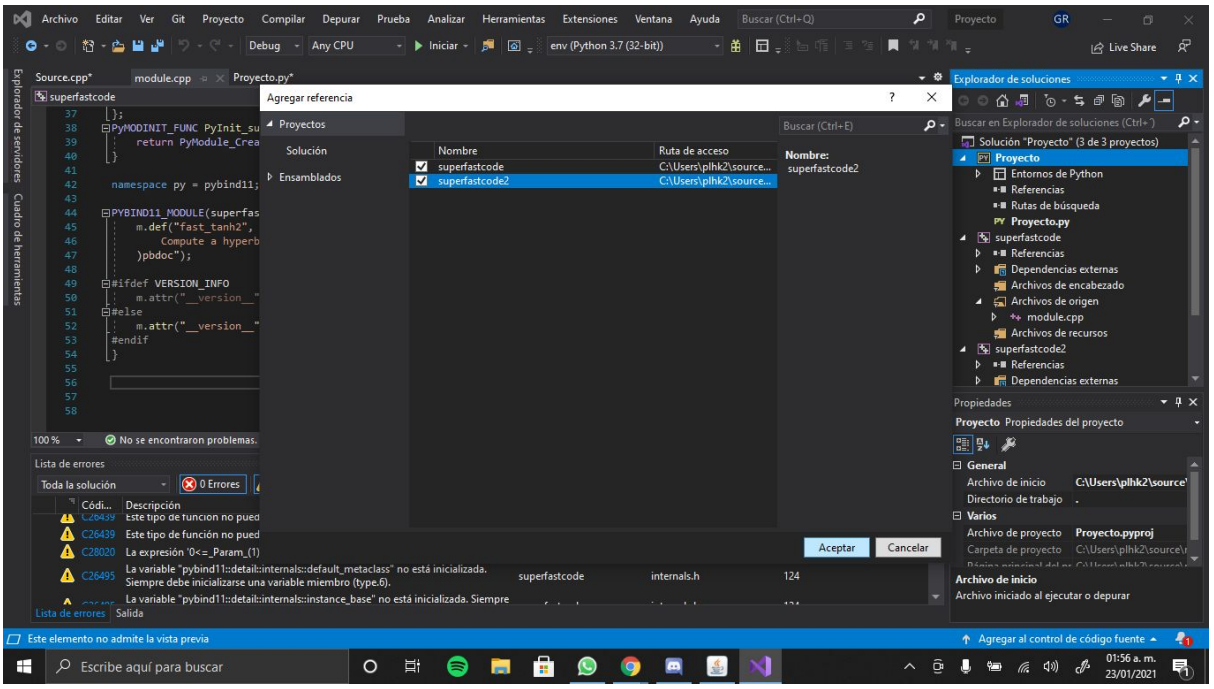

};
PyMODINIT_FUNC PyInit_superfastcode() {
return PyModule_Create(&superfastcode_module);
}
namespace py = pybind11;
PYBIND11_MODULE(superfastcode2, m) {
m.def("fast_tanh2", &tanh_impl, R"pbdoc(
Compute a hyperbolic tangent of a single argument expressed in radians.
)pbdoc");
#ifdef VERSION_INFO
m.attr("__version__") = VERSION_INFO;
#else
m.attr("__version__") = "dev";
#endif
}
100 %
No se encontraron problemas.
Línea: 56 Carácter: 1 SPC CRLF
Salida
Mostrar salida de: Compilación
Lista de errores Salida
Explorador de soluciones
Buscar en Explorador de soluciones (Ctrl+;)
Explorador de soluciones Cambios de GIT
Propiedades
Listo
Agregar al control de código fuente
Escribe aquí para buscar
01:47 a. m.
23/01/2021

1- Probar el código y comparar los resultados Ahora que tiene los archivos DLL estructurados como extensiones de Python, puede hacerles referencia desde el proyecto de Python, importar los módulos y usar sus métodos.

2- Poner el archivo DLL a disposición de Python Hay dos maneras de hacer que el archivo DLL esté disponible para Python. El primer método funciona si el proyecto de Python y el de C++ se encuentran en la misma solución.

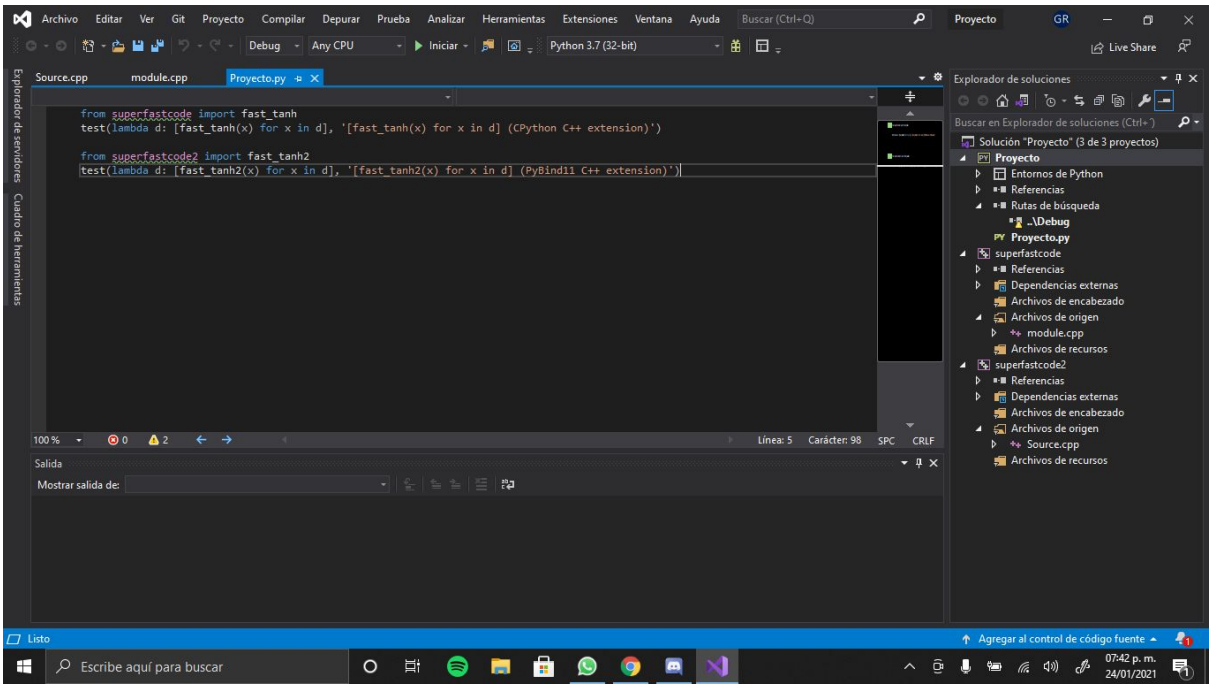
- Vaya al Explorador de soluciones, haga clic con el botón derecho en el nodo Referencias del proyecto de Python y, después, seleccione Agregar referencia. En el cuadro de diálogo que aparecerá, seleccione la pestaña Proyectos, los proyectos superfastcode y superfastcode2 y Aceptar.



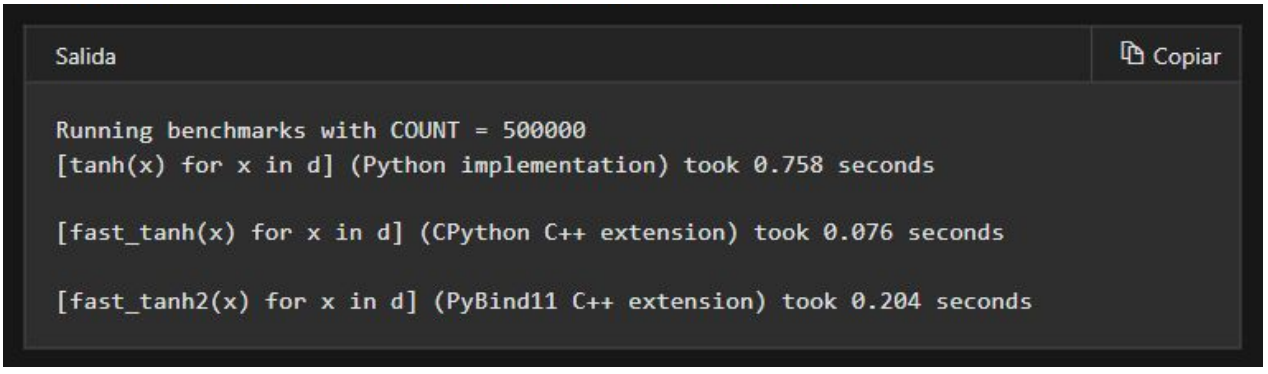


Llamar al archivo DLL desde Python Una vez que el archivo DLL está disponible para Python como se ha descrito en la sección anterior, ahora puede llamar a las funciones superfastcode.fast_tanh y superfastcode2.fast_tanh2 del código de Python y comparar su rendimiento con la implementación de Python:

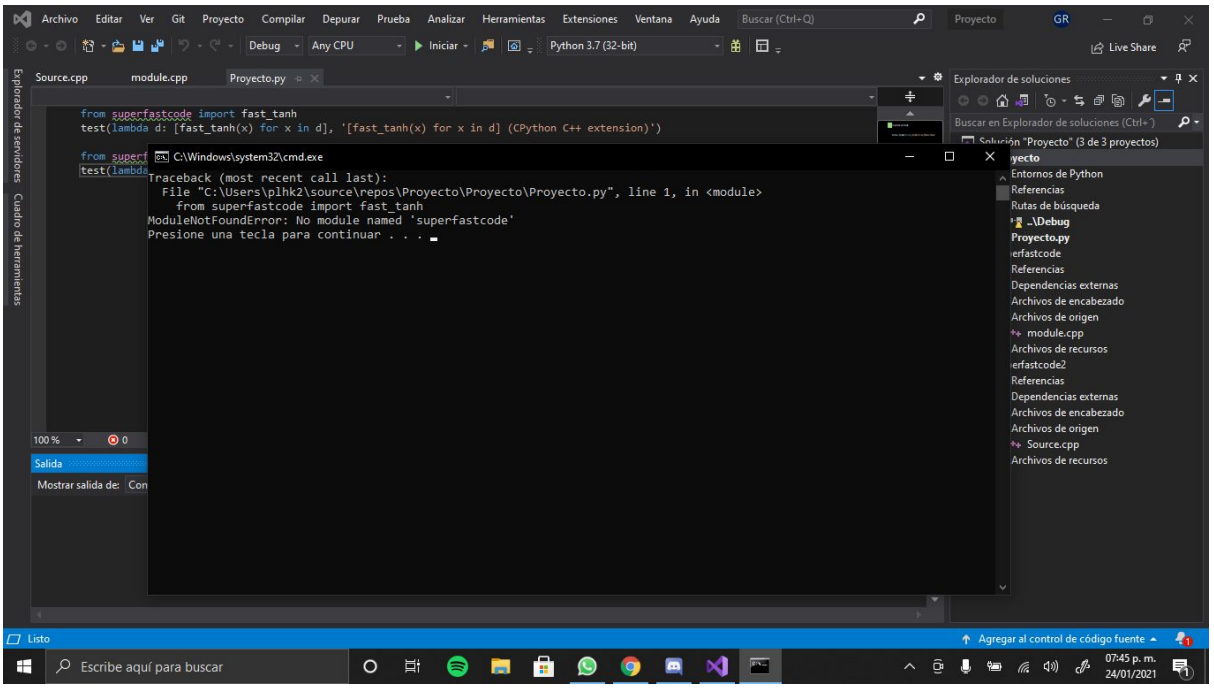
1- Agregue las líneas siguientes en el archivo .py para llamar a los métodos exportados desde los archivos DLL y mostrar los resultados.



Ejecute el programa Python (Depurar > Iniciar sin depurar o Ctrl + F5) y observe que las rutinas de C ++ se ejecutan aproximadamente de cinco a veinte veces más rápido que la implementación de Python. La salida típica aparece como sigue:



En nuestro caso nos salió un resultado diferente ya que nos muestra la siguiente ventana



Al revisar las soluciones encontramos que en la página la solución mencionada requiere de Visual Studio de versión anterior puesto que la versión que tenemos es la más reciente(2019) y el post del que nos basamos para comprobar dicha conexión no está actualizado para esta versión no nos fue posible demostrar la conexión entre dos lenguajes de programación.

Lo que notamos en el desarrollo de este proyecto fue que una de las ventajas que tiene esta nueva librería que nos asigno el profesor es la compatibilidad con otro lenguaje de programación ya que cuenta con nuevas librerías dentro de ella que se lo permiten y son más rápidas.

Requerimientos funcionales y no funcionales:

Funcionales:

- Mejor compatibilidad con IDE en Visual Studio
- Editar, compilar y depurar proyectos de C ++ Visual Studio
- Mejora el rendimiento en el ambiente de desarrollo c++ Visual studio

No funcionales:

- Ejecución rápida.
- Corrección según lo definido por el estándar C ++ 11.
- Tiempos de compilación rápidos.
- Uso mínimo de memoria.

FUENTES:

Anonimo. (s/f). Biblioteca estándar de C ++ "libc ++". Diciembre 2020, de LLVM Sitio web: https://libcxx.llvm.org/

Anonimo. (s/f). documentación de libc ++ 8.0. Diciembre 03, 2020, de bcain Sitio web: https://bcain-llvm.readthedocs.io/projects/libcxx/en/latest/

Meza Juan D.(2020). Bibliotecas o librerías en C++. Declaración y uso de librerías. Include en C++. Recuperado el 10/12/2020, del sitio web https://www.programarya.com/Cursos/C++/Bibliotecas-o-Librerias

Anónimo (2018). Creación de una extensión de C++ para Python. Recuperado el 12/12/2020, del sitio https://docs.microsoft.com/es-es/visualstudio/python/working-with-c-cpp-python-in-visual-studio?view=vs-2019

Anónimo (2020). Compatibilidad con Clang / LLVM en proyectos de Visual Studio. Recuperado el 12/12/2020. del sitio https://docs.microsoft.com/en-us/cpp/build/clang-support-msbuild?view=msvc-160

Emanuel (2020).¿Clang ++ ABI es lo mismo que g ++?. Recuperado el 8/01/2021 del sitio https://www.javaer101.com/es/article/2948533.html

Anónimo . (20/04/2020). Compilación y ejecución de un proyecto de aplicación de consola de C++. 22/01/2021, de Micrososft Sitio web: https://docs.microsoft.com/es-es/cpp/build/vscpp-step-2-build?view=msvc-160

Anónimo. (20/04/2020). Creación de un proyecto de aplicación de consola de C++. 22/01/2021, de Microsoft Sitio web: https://docs.microsoft.com/es-es/cpp/build/vscpp-step-1-create?view=msvc-160

Anónimo . (05/11/2020). Instalación de compatibilidad con C y C++ en Visual Studio. 22/01/2021, de Microsoft Sitio web: https://docs.microsoft.com/es-es/cpp/build/vscpp-step-0-installation?view=msvc-160

Anónimo . (20/04/2020). Creación de un proyecto de aplicación de consola de C++. 22/01/201, de Microsoft Sitio web: https://docs.microsoft.com/es-es/cpp/build/vscpp-step-1-create?view=msvc-160