User

I cannot install or compile rdrand library of python on Windows and pip install . command in folder where it is installed always asks for visual studio which I do not want to use. I want to use gcc which it is not allowing for compilation.

Under the circumstances, I was exploring other optional libraries in python which use intel intrinsic function rdrand and which can be easily used in windows. Can you find out the other alternatives in Python (Not C)

Bing

I understand that you're having trouble installing the `rdrand` library on Windows using pip and you're looking for alternatives. Unfortunately, the `rdrand` library is one of the few that provides a Python interface to Intel's hardware random number generator (RNG) using the RdRand instruction.

However, you might consider using Python's built-in `random` module or `numpy.random` if you're working with NumPy. These modules use software-based RNGs and should work on any system where Python is installed. They might not provide the same level of randomness as a hardware RNG, but they are often sufficient for many applications.

If you specifically need a hardware RNG, you might need to consider using a different programming language that has better support for Intel's RdRand instruction on Windows. For example, C and C++ have intrinsic support for RdRand.

Please note that using hardware RNGs can have implications for the security and predictability of the random numbers generated. Always consider the requirements of your specific application when choosing a RNG. If you have any other questions or need further assistance, feel free to ask! 😊

User

As per my recent exploration RandomGen is another library with backend as rdrand intel intrinsic. but it has been merged in numpy. So whether numpy is able to generate True Random Numbers (TRNG) basedon rdrand as back end? If so, what is the use of having a dedicated library rdrand in python?

Bing

Yes, you're correct. The `RandomGen` library, which had `RDRAND` as one of its backends, has been merged into `NumPy`. This means that you can use `NumPy` to generate random numbers using the `RDRAND` instruction.

The `RDRAND` instruction is a hardware-based random number generator (RNG) provided by Intel. It's considered a True Random Number Generator (TRNG) because it uses physical processes (like electronic noise) to generate random numbers, rather than a deterministic algorithm.

As for why there's a dedicated `rdrand` library in Python, it's likely for convenience and ease of use. The `rdrand` library provides a simple, straightforward interface to the `RDRAND` instruction. If you're writing a program that needs to generate random numbers and you want to take advantage of the `RDRAND` instruction, using the `rdrand` library might be easier than using `NumPy`.

However, if you're already using `NumPy` for other parts of your program, or if you need to generate large arrays of random numbers, using `NumPy` might be more convenient. Plus, `NumPy` provides many other features and functions for numerical computing, so it's a more versatile library overall.

In conclusion, whether you should use `rdrand` or `NumPy` depends on your specific needs and the context of your project. Both libraries provide access to the `RDRAND` instruction, but they offer different interfaces and additional features. I hope this helps! If you have any other questions, feel free to ask. 😊

User

How to access rdrand intel intrinsic using numpy?

Bing

While `NumPy` does not directly expose the `RDRAND` instruction, it does use a variety of random number generation methods in its `random` module. However, there is no explicit way to select `RDRAND` as the random number generator in

`NumPy`.

The `rdrand` library in Python provides a more direct interface to the `RDRAND` instruction. It’s a subclass of `random.Random` and behaves like `random.Random`, but it uses inline assembly to access the hardware RNG using the `RdRand` instruction. This should be a cryptographically secure drop-in replacement for `random` with a prediction complexity bound of `O(2^128)`, if the Intel random number generator is valid.

If you’re looking for a way to generate random numbers in Python using `RDRAND`, you might need to use the `rdrand` library or a similar library that provides a Python interface to the `RDRAND` instruction.

Please note that the use of hardware RNGs can have implications for the security and predictability of the random numbers generated. Always consider the requirements of your specific application when choosing a RNG. If you have any other questions or need further assistance, feel free to ask! 😊

User

What do I need to change in setup.py to compile with gcc instead of visual studio?

Bing

To compile with `gcc` instead of `Visual Studio`, you can try setting the compiler name by specifying a `CC` environment variable in your `setup.py` file. Here’s how you can do it:

```python
import os

os.environ["CC"] = "gcc"
os.environ["CXX"] = "g++"

# rest of your setup.py code
```

In this code, `os.environ["CC"] = "gcc"` sets the C compiler to `gcc`, and `os.environ["CXX"] = "g++"` sets the C++ compiler to `g++`.

Alternatively, you can provide default command line arguments for distutils in a separate file called `setup.cfg` (placed parallel to your `setup.py`). To set the compiler, you can use something like this:

```ini
[build]
compiler=mingw32
```

In this configuration, `compiler=mingw32` sets the compiler to MinGW, which is a minimalist development environment for native Microsoft Windows applications.

Please replace `"gcc"` and `"g++"` or `"mingw32"` with the actual paths to your `gcc`, `g++`, or `MinGW` installations if they’re not in your system’s `PATH`.

Remember to save your changes and rerun your `setup.py` script. If you have any other questions or need further assistance, feel free to ask! 😊

User

My setup.py is as under:

from distutils.core import setup, Extension os.environ[“CC”] = “gcc” os.environ[“CXX”] = “g++” setup(name=‘rdrand’, version=‘1.4.9’, description=“Python interface to Intel hardware rng”, long_description= “”.join(open(‘rdrandom.rst’).readlines()), author=“Chris Stillson”, author_email=“stillson@gmail.com”, url=‘https://github.com/stillson/rdrand’, license=“New BSD license”, ext_modules=[Extension(‘_rdrand’, [‘rdrand.c’])], py_modules = [‘rdrand’], keywords = [‘rdrand’, ‘rdseed’, “intel”,“hardware”,“random”,“number”,“generator”,“rng”], data_files=[(‘share’, [‘rdrandom.rst’])], classifiers = [“Development Status :: 4 - Beta”, “Intended Audience :: Developers”, “Intended Audience :: Science/Research”, “License :: OSI Approved :: BSD License”, “Programming Language :: Python”, “Programming Language :: Python :: 2”, “Programming Language :: Python :: 3”, “Topic :: Security”, “Topic :: Security :: Cryptography”,], extras_require={ ‘tests’: [], ‘docs’: [‘sphinx >= 1.4’, ‘sphinx_rtd_theme’]}) Since distutils has been deprecated in python 3.10 what change is required in above? Further do I need to include extension command for compiling through gcc?

Bing

You’re correct that `distutils` has been deprecated in Python 3.10. The recommended replacement for `distutils` is `setuptools`. Here’s how you can modify your `setup.py` to use `setuptools` instead:

```python
from setuptools import setup, Extension
import os

os.environ["CC"] = "gcc"
os.environ["CXX"] = "g++"

setup(
    name='rdrand',
    version='1.4.9',
    description="Python interface to Intel hardware rng",
    long_description= "".join(open('rdrandom.rst').readlines()),
    author="Chris Stillson",
    author_email="stillson@gmail.com",
    url='https://github.com/stillson/rdrand',
    license="New BSD license",
    ext_modules=[Extension('_rdrand', ['rdrand.c'])],
    py_modules = ['rdrand'],
    keywords = ['rdrand', 'rdseed', "intel","hardware","random","number","generator","rng"],
    data_files=[('share', ['rdrandom.rst'])],
    classifiers = ["Development Status :: 4 - Beta",
        "Intended Audience :: Developers",
        "Intended Audience :: Science/Research",
        "License :: OSI Approved :: BSD License",
        "Programming Language :: Python",
        "Programming Language :: Python :: 2",
        "Programming Language :: Python :: 3",
        "Topic :: Security",
        "Topic :: Security :: Cryptography",],
    extras_require={
        'tests': [],
        'docs': [
            'sphinx >= 1.4',
            'sphinx_rtd_theme']}
)
```

In this code, I’ve replaced `from distutils.core import setup, Extension` with `from setuptools import setup, Extension`.

As for your question about the `Extension` command, it’s used to compile C and C++ extensions for Python. The `Extension` command takes two arguments: the name of the extension module, and a list of source filenames. It doesn’t directly control which compiler is used; that’s determined by the `CC` and `CXX` environment variables.

I hope this helps! If you have any other questions, feel free to ask. 😊

User

following is the output whil compiling using pip install . in directory d:\downloads\rdrand:

PS D:\Downloads\rdrand> pip install . Processing d:\downloads\rdrand Preparing metadata (setup.py) … done Building wheels for collected packages: rdrand Building wheel for rdrand (setup.py) … error error: subprocess-exited-with-error

× python setup.py bdist_wheel did not run successfully. │ exit code: 1 ╰─> [9 lines of output] running bdist_wheel running build running build_py creating build creating build\lib.win-amd64-cpython-312 copying rdrand.py -> build\lib.win-amd64-cpython-312 running build_ext building ‘_rdrand’ extension error: Microsoft Visual C++ 14.0 or greater is required. Get it with “Microsoft C++ Build Tools”: https://visualstudio.microsoft.com/visual-cpp-build-tools/ [end of output]

note: This error originates from a subprocess, and is likely not a problem with pip. ERROR: Failed building wheel for rdrand Running setup.py clean for rdrand Failed to build rdrand ERROR: Could not build wheels for rdrand, which is required to install pyproject.toml-based projects

Bing

The error message you’re seeing is indicating that Microsoft Visual C++ 14.0 or greater is required. This is a common issue when trying to install certain Python packages on Windows.

Here are some steps you can follow to resolve this issue:

1. Download and install the latest version of Microsoft C++ Build Tools.
2. During the installation, make sure to select the following workloads:
    - **C++ build tools**
    - **Windows 10 SDK**
3. After the installation is complete, restart your computer.
4. Upgrade pip to the latest version using the command: `python -m pip install --upgrade pip`.
5. Try running your `pip install .` command again.

If you've already installed Microsoft Visual C++ 14.0 and are still seeing this error, it's possible that the installation didn't complete successfully or the necessary components weren't selected during the installation. In that case, you might need to reinstall it and make sure to select the correct components.

I hope this helps! If you have any other questions, feel free to ask. 😊.

User

But I want to build using gcc. That's why I gave my setup.py file as above. Suppose I use:

python setup.py build -c mingw32

and then pip intall .

Will it compile?

Bing

Yes, you can build the extension with `gcc` using the command `python setup.py build -c mingw32`. This command tells `setup.py` to build the extension using the `mingw32` compiler, which is a port of `gcc`.

After running this command, you should be able to install the package with `pip install .`. This command tells `pip` to install the package from the current directory.

However, please note that this process might not work if the `setup.py` script is not properly configured to use `gcc` or if there are issues with your `gcc` installation. If you encounter any errors, please let me know and I'll do my best to help you troubleshoot them.

I hope this helps! If you have any other questions, feel free to ask. 😊

User

I could make rdrand compiled without visual studio on windows - finally.

Bing

That's great news! I'm glad to hear that you were able to compile the `rdrand` library without using Visual Studio on Windows. It's a testament to your perseverance and problem-solving skills. If you have any more questions or need further assistance, feel free to ask. 😊

User

Now I want to use this library to generate random numbers. I wish to use xlwings so the output shall be list of list. Can you give me python script to do this?

Bing

Sure, I can help with that. Here's a simple Python script that uses the `rdrand` library to generate random numbers and formats the output as a list of lists. This script assumes that you want to generate a 2D array of random numbers.

```
import rdrand
import xlwings as xw

# Create an instance of the rdrand.RdRandom class
rng = rdrand.RdRandom()

# Define the dimensions of your 2D array
num_rows = 5
num_cols = 5

# Generate the random numbers
```

```
random_numbers = [[rng.random() for _ in range(num_cols)] for _ in range(num_rows)]

# Output the random numbers to Excel using xlwings
wb = xw.Book()  # This will create a new Excel workbook
sheet = wb.sheets[0]  # This will select the first (and only) sheet in the workbook
sheet.range('A1').value = random_numbers  # This will write the random numbers to the sheet starting at cell A1
```

In this script, `rng.random()` generates a random floating-point number between 0 and 1. The list comprehension `[[rng.random() for _ in range(num_cols)] for _ in range(num_rows)]` generates a 2D array (a list of lists) of these random numbers.

The `xlwings` part of the script writes this 2D array to an Excel workbook. `xw.Book()` creates a new workbook, `wb.sheets[0]` selects the first sheet in the workbook, and `sheet.range('A1').value = random_numbers` writes the random numbers to the sheet starting at cell A1.

You can adjust `num_rows` and `num_cols` to generate a 2D array of any size. You can also adjust the `sheet.range('A1').value = random_numbers` line to write the random numbers to a different range in the sheet.

I hope this helps! If you have any other questions, feel free to ask. 😊

User

I want to generate using python UDF. Input will be number of random numbers to be generated and output will be one columner excel column (as list of list).

Bing

Sure, you can create a Python User-Defined Function (UDF) with `xlwings` to generate random numbers using the `rdrand` library. Here's an example of how you can do this:

```
import rdrand
import xlwings as xw
from xlwings import utils

# Create an instance of the rdrand.RdRandom class
rng = rdrand.RdRandom()

@xw.func
def generate_random_numbers(num):
    # Generate the random numbers
    random_numbers = [[rng.random()] for _ in range(num)]
    return random_numbers

@xw.sub
def main():
    # Expose the UDF to Excel
    utils.excel.register_udfs([generate_random_numbers])

# Run the main function when the script is executed
if __name__ == "__main__":
    xw.Book().set_mock_caller()
    main()
```

In this script, `generate_random_numbers` is a UDF that generates a specified number of random numbers and returns them as a list of lists (each sublist contains one number). The `@xw.func` decorator tells `xlwings` that this function is a UDF that should be exposed to Excel.

The `main` function uses `utils.excel.register_udfs` to register the UDF with Excel. The `@xw.sub` decorator tells `xlwings` that this function is a subroutine that should be run when the script is executed.

The `if __name__ == "__main__":` block at the end of the script ensures that the `main` function is run when the script is executed.

Once you've created this script, you can use the `generate_random_numbers` function in Excel just like any other Excel function. For example, you could enter `=generate_random_numbers(5)` into a cell to generate 5 random numbers.

I hope this helps! If you have any other questions, feel free to ask. 😊

User

traceback:

pythoncom error: Python error invoking COM method.

Traceback (most recent call last): File “D:\Programs\Python\Lib\site-packages\win32com\server\policy.py”, line 305, in Invoke return self.invoke(dispid, lcid, wFlags, args) ^^^ File “D:\Programs\Python\Lib\site-packages\win32com\server\policy.py”, line 310, in invoke return S_OK, -1, self.invokeex(dispid, lcid, wFlags, args, None, None) ^^^ File “D:\Programs\Python\Lib\site-packages\win32com\server\policy.py”, line 639, in invokeex return func(*args) ^^^^^^^^^^^ File “D:\Programs\Python\Lib\site-packages\xlwings\com_server.py”, line 235, in CallUDF res = call_udf(script, fname, args, this_workbook, FromVariant(caller)) ^^^ File “D:\Programs\Python\Lib\site-packages\xlwings\udfs.py”, line 525, in call_udf ret = func(*args) ^^^^^^^^^^^ File “D:\Onedrive\OneDrive - 0yt2k\xlwings_trial.py”, line 151, in generate_random_numbers_rdrand random_numbers = [[rng.random()] for _ in range(num)] ^^^^^^^^^^ TypeError: ‘float’ object cannot be interpreted as an integer 2023-11-21 10:04:24,610 - distributed.batched - INFO - Batched Comm Closed <InProc (closed) Client->Scheduler local=inproc://192.168.1.6/12368/8 remote=inproc://192.168.1.6/12368/1> Traceback (most recent call last): File “D:\Programs\Python\Lib\site-packages\distributed\batched.py”, line 115, in _background_send nbytes = yield coro ^^^^^^^^^^ File “D:\Programs\Python\Lib\site-packages\tornado\gen.py”, line 767, in run value = future.result() ^^^^^^^^^^^^^^^ File “D:\Programs\Python\Lib\site-packages\distributed\comm\inproc.py”, line 223, in write raise CommClosedError() distributed.comm.core.CommClosedError

Bing

The error message you’re seeing, `TypeError: 'float' object cannot be interpreted as an integer`, is typically raised when a floating-point number (a number with a decimal point) is used in a context that requires an integer.

In your case, the error is occurring on this line:

```python
random_numbers = [[rng.random()] for _ in range(num)]
```

The `range()` function requires an integer argument, but it seems that `num` is a floating-point number.

To fix this error, you can convert `num` to an integer before passing it to the `range()` function. Here’s how you can do it:

```python
random_numbers = [[rng.random()] for _ in range(int(num))]
```

In this code, `int(num)` converts `num` to an integer. This should resolve the error and allow your script to run successfully.

I hope this helps! If you have any other questions, feel free to ask. 😊

User

Numbers are generated like this:

0.974394632 0.613978549 0.824866877 0.60333449 0.186893559 0.958721293 0.114839648 0.134841548 0.940713782

Instead of like this when I used C compiled rdrand:

492466049003309 156740128279014 640830463744412 964380657606811 659400858928961 167373430593139 455566206717995 157429368503989 340547383809323 583676626259486

Bing

The `rdrand` function in Python’s `rdrand` library returns a floating-point number between 0 and 1. This is why you’re seeing numbers like `0.974394632`, `0.613978549`, etc.

On the other hand, when you used the `RDRAND` instruction in C, you likely accessed the raw 64-bit integer that `RDRAND` generates. This is why you’re seeing larger numbers like `492466049003309`, `156740128279014`, etc.

If you want to generate similar large random numbers in Python, you can multiply the output of `rng.random()` by a large number, such as `10**15`, and then convert it to an integer. Here’s how you can modify the `generate_random_numbers` function to do this:

```python
@xw.func
def generate_random_numbers(num):
    # Generate the random numbers
    random_numbers = [[int(rng.random() * 10**15)] for _ in range(int(num))]
    return random_numbers
```

In this code, `rng.random() * 10**15` generates a large floating-point number, and `int(rng.random() * 10**15)` converts it to an integer.

I hope this helps! If you have any other questions, feel free to ask. 😊

User

Can you further modify so as to generate each number as fixed 15 digits integers

Bing

Sure, you can modify the `generate_random_numbers` function to generate 15-digit integers. Here's how you can do it:

```
@xw.func
def generate_random_numbers(num):
    # Generate the random numbers
    random_numbers = [[int(rng.random() * 10**14) + 10**14] for _ in range(int(num))]
    return random_numbers
```

In this code, `rng.random() * 10**14` generates a 14-digit floating-point number, and `int(rng.random() * 10**14) + 10**14` converts it to a 15-digit integer. The `+ 10**14` ensures that the integer always has 15 digits.

Please note that this method might not generate all possible 15-digit integers with equal probability, because the `rng.random()` function generates floating-point numbers between 0 and 1, not integers. If you need to generate all possible 15-digit integers with equal probability, you might need to use a different method.

I hope this helps! If you have any other questions, feel free to ask. 😊

User

How do I generate random numbers with a specific range in Python?

Bing

In Python, you can use the `random.randint(a, b)` function from the `random` module to generate random integers within a specific range. Here's an example:

```
import random

# Generate a random integer between 10 and 20
random_number = random.randint(10, 20)

print(random_number)
```

In this code, `random.randint(10, 20)` generates a random integer between 10 and 20, inclusive.

If you want to generate a random floating-point number within a specific range, you can use the `random.uniform(a, b)` function. Here's an example:

```
import random

# Generate a random floating-point number between 0.5 and 1.5
random_number = random.uniform(0.5, 1.5)

print(random_number)
```

In this code, `random.uniform(0.5, 1.5)` generates a random floating-point number between 0.5 and 1.5.

I hope this helps! If you have any other questions, feel free to ask. 😊

User

I am not able to export this chat to word. I clicked on three dots on right of chat heading gices option as edit in word which when clicked leads me to onedrive and nothing happens.

Bing

I understand that you’re having trouble exporting a Teams chat to Word. Based on the information I found, there are several methods you can try:

1. **Copy and Paste**: You can manually copy the messages and paste them into a Word document.
2. **Screenshots**: You can take screenshots of the conversation and insert them into a Word document.
3. **Share to Outlook**: You can share a single chat to Outlook and then save it as a Word file.
4. **Export Page**: You can launch the Export Page in your Teams account and select Chat history.
5. **Print as PDF**: You can open Teams in a browser and select Print as PDF, then convert the PDF to Word.
6. **Pop Out Conversation**: You can pop up the conversation from Teams client and copy and paste it into a Word document.

Please note that these methods might not work in all situations. For example, some users have reported that the “Export chat history” option is not available. If you’re still having trouble, you might want to contact Microsoft Support for further assistance.

I hope this helps! If you have any other questions, feel free to ask. 😊

User

I am not talking about teams chat. I refer to chat here - bing AI.

Bing

I’m sorry for the misunderstanding. As of my knowledge up to 2021, there isn’t a built-in feature to export this chat directly to a Word document. However, you can manually copy the text from the chat and paste it into a Word document. If you want to include the timestamps or other details, you might need to copy those separately. I hope this helps! If you have any other questions, feel free to ask. 😊