```python
def main():
    while True:
        au = input("AU: ")
        try:
            au = float(au)
            break
        except ValueError:
            continue

    print(f"{au} AU is {convert(au)} m")


def convert(au):
    return au * 149597870700


if __name__ == "__main__":
    main()
```

```python
from convert0 import convert


def test_conversion():
    assert convert(1) == 149597870700
    assert convert(50) == 7479893535000
```

```python
def main():
    while True:
        au = input("AU: ")
        try:
            au = float(au)
            break
        except ValueError:
            continue

    print(f"{au} AU is {convert(au)} m")


def convert(au):
    if not isinstance(au, (int, float)):
        raise TypeError("au must be an int or float")
    return au * 149597870700


if __name__ == "__main__":
    main()
```

```python
import pytest
from convert1 import convert


def test_conversion():
    assert convert(1) == 149597870700
    assert convert(50) == 7479893535000


def test_error():
    with pytest.raises(TypeError):
        convert("1")
```

```python
import pytest
from convert1 import convert


def test_int_conversion():
    assert convert(1) == 149597870700
    assert convert(50) == 7479893535000


def test_error():
    with pytest.raises(TypeError):
        convert("1")


def test_float_conversion():
    assert convert(0.001) == pytest.approx(149597870.691)
```

```python
import pytest
from convert1 import convert


def test_int_conversion():
    assert convert(1) == 149597870700
    assert convert(50) == 7479893535000


def test_error():
    with pytest.raises(TypeError):
        convert("1")


def test_float_conversion():
    assert convert(0.001) == pytest.approx(149597870.691, abs=0.1)
```

```python
import pytest
from convert1 import convert


def test_int_conversion():
    assert convert(1) == 149597870700
    assert convert(50) == 7479893535000


def test_error():
    with pytest.raises(TypeError):
        convert("1")


def test_float_conversion():
    assert convert(0.001) == pytest.approx(149597870.691, abs=1e-12)
```