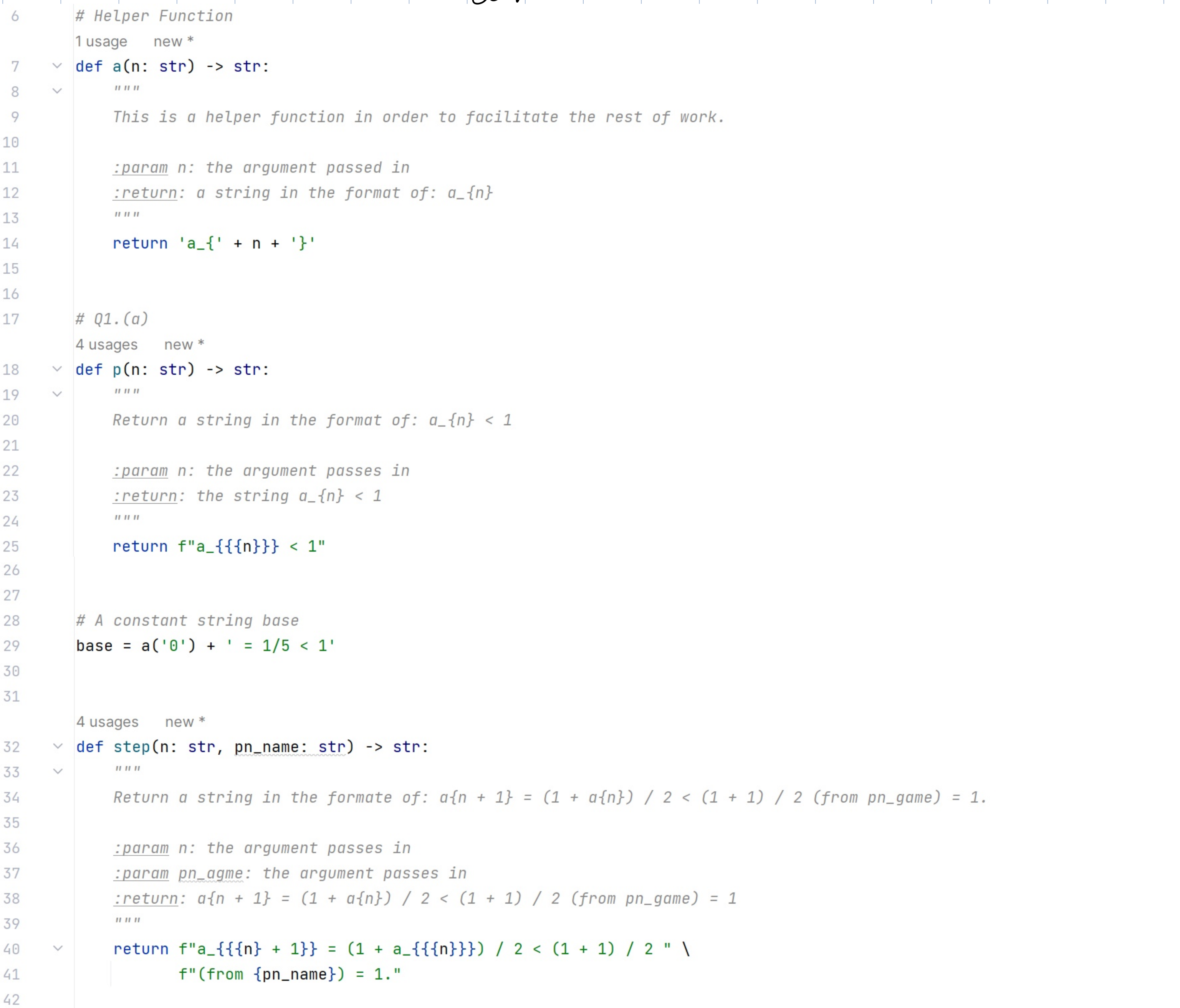
Problem Set 3.

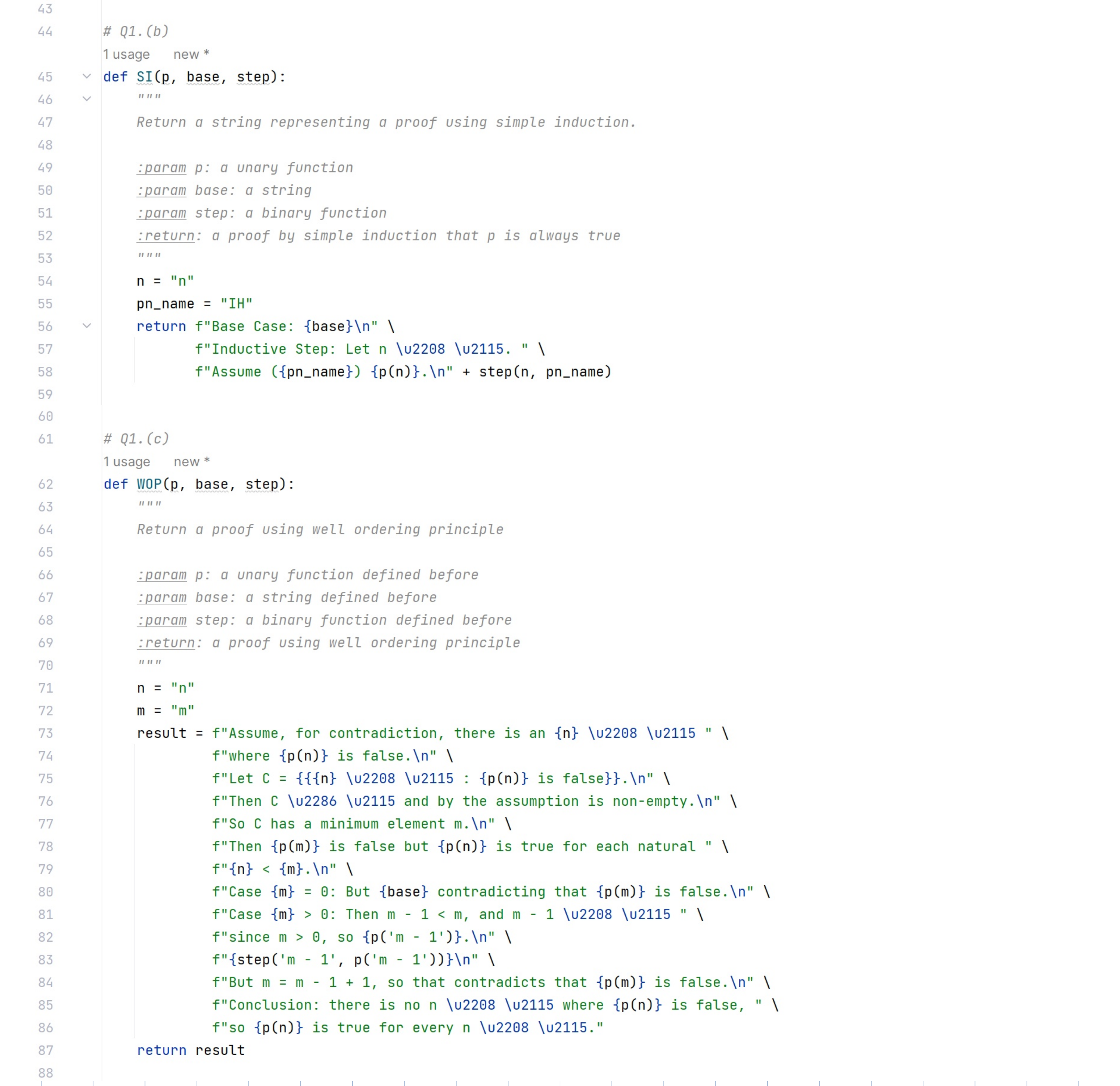
Xuanqi Wei

1009353209

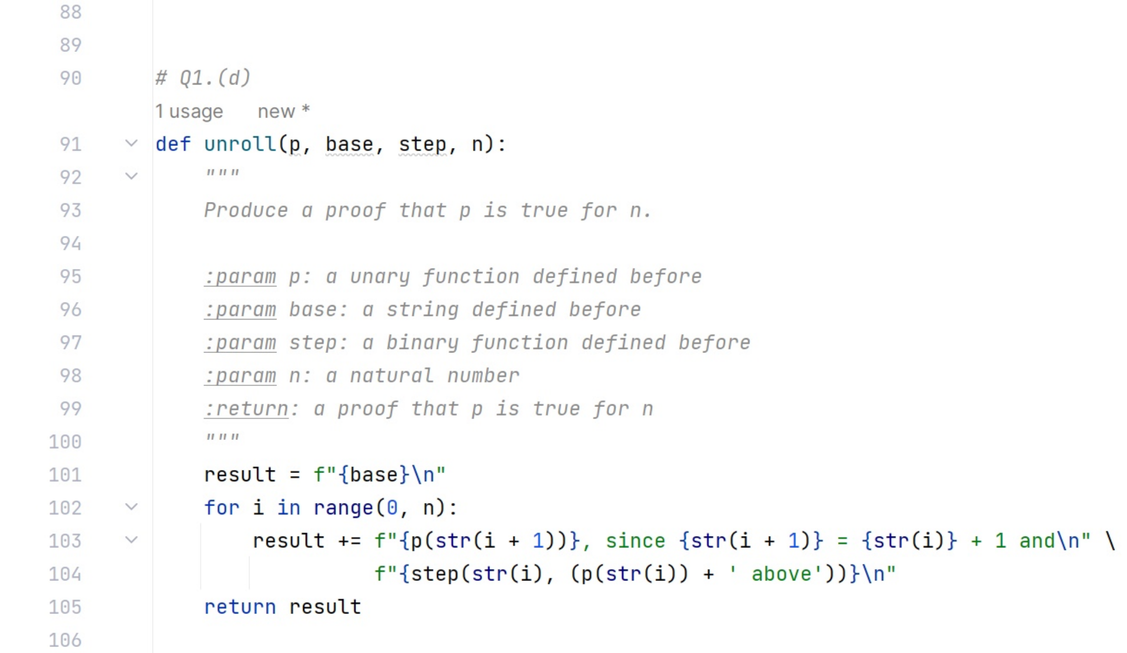
Q1.



```python
# Helper Function
def a(n: str) -> str:
    """
    This is a helper function in order to facilitate the rest of work.

    :param n: the argument passed in
    :return: a string in the format of: a_{n}
    """
    return 'a_{' + n + '}'


# Q1.(a)
def p(n: str) -> str:
    """
    Return a string in the format of: a_{n} < 1

    :param n: the argument passes in
    :return: the string a_{n} < 1
    """
    return f"a_{{{n}}} < 1"


# A constant string base
base = a('0') + ' = 1/5 < 1'


def step(n: str, pn_name: str) -> str:
    """
    Return a string in the formate of: a{n + 1} = (1 + a{n}) / 2 < (1 + 1) / 2 (from pn_game) = 1.

    :param n: the argument passes in
    :param pn_agme: the argument passes in
    :return: a{n + 1} = (1 + a{n}) / 2 < (1 + 1) / 2 (from pn_game) = 1
    """
    return f"a_{{{n} + 1}} = (1 + a_{{{n}}}) / 2 < (1 + 1) / 2 " \
           f"(from {pn_name}) = 1."

```



```python

# Q1.(b)
1 usage   new *
def SI(p, base, step):
    """
    Return a string representing a proof using simple induction.

    :param p: a unary function
    :param base: a string
    :param step: a binary function
    :return: a proof by simple induction that p is always true
    """
    n = "n"
    pn_name = "IH"
    return f"Base Case: {base}\n" \
           f"Inductive Step: Let n \u2208 \u2115. " \
           f"Assume ({pn_name}) {p(n)}.\n" + step(n, pn_name)


# Q1.(c)
1 usage   new *
def WOP(p, base, step):
    """
    Return a proof using well ordering principle

    :param p: a unary function defined before
    :param base: a string defined before
    :param step: a binary function defined before
    :return: a proof using well ordering principle
    """
    n = "n"
    m = "m"
    result = f"Assume, for contradiction, there is an {n} \u2208 \u2115 " \
             f"where {p(n)} is false.\n" \
             f"Let C = {{{n} \u2208 \u2115 : {p(n)} is false}}.\n" \
             f"Then C \u2286 \u2115 and by the assumption is non-empty.\n" \
             f"So C has a minimum element m.\n" \
             f"Then {p(m)} is false but {p(n)} is true for each natural " \
             f"{n} < {m}.\n" \
             f"Case {m} = 0: But {base} contradicting that {p(m)} is false.\n" \
             f"Case {m} > 0: Then m - 1 < m, and m - 1 \u2208 \u2115 " \
             f"since m > 0, so {p('m - 1')}.\n" \
             f"{step('m - 1', p('m - 1'))}\n" \
             f"But m = m - 1 + 1, so that contradicts that {p(m)} is false.\n" \
             f"Conclusion: there is no n \u2208 \u2115 where {p(n)} is false, " \
             f"so {p(n)} is true for every n \u2208 \u2115."
    return result

```



```python
# Q1.(d)
def unroll(p, base, step, n):
    """
    Produce a proof that p is true for n.

    :param p: a unary function defined before
    :param base: a string defined before
    :param step: a binary function defined before
    :param n: a natural number
    :return: a proof that p is true for n
    """
    result = f"{base}\n"
    for i in range(0, n):
        result += f"{p(str(i + 1))}, since {str(i + 1)} = {str(i)} + 1 and\n" \
                  f"{step(str(i), (p(str(i)) + ' above'))}\n"
    return result
```