Lab 3 Task 8 Solution

8) Additional Tasks

8.1) A user player

```
    ...
    class UserPlayer(Player):

    def move(self, current: int, min_step: int,
             max_step: int, goal: int) -> int:

        amount = 0

        while True:
            amount_raw = input('Enter step amount ({}-{})'.format(min_step
    , max_step))

            if len(amount_raw.strip()) == 0:
                print('Please select integer value between {} and {}'.
    format(min_step, max_step))
                continue

            if re.search(r'[^0-9]+', amount_raw):
                print('Please select integer value between {} and {}'.
    format(min_step, max_step))
                continue

            amount = int(amount_raw)
            if amount < min_step or amount > max_step:
                print('Please select steps between {} and {}'.format(
    min_step, max_step))
                continue

            break

        return amount

    ...
    def make_player(generic_name: str) -> Player:
        ...
        return UserPlayer(name)

```

```
    ...

    if __name__ == '__main__':
        # Uncomment the lines below to check your work using
        # python_ta and doctest.
        # import python_ta
        # python_ta.check_all(config={
        #     'extra-imports': ['random'],
        #     'allowed-io': [
        #         'main',
        #         'make_player',
        #         'move',
        #         'play_one_turn'
        #     ]
        # })
        main()
```

8.2) A strategic player

The solution to this problem makes following assumptions:

- *goal* of 21
- *min_step* of 1
- *max_step* of 3
- one of the player as *StrategicPlayer*
- the other as *RandomPlayer*

We need to create *StrategicPlayer* that always wins as player 1, and does win as player 2 when a bad move is by the other player. Also, we need to adjust *make_player* so a player's type can be chosen by user.

```
    ...

    # ================ SOLUTION (Task 8.2) ================

    class StrategicPlayer(Player):

        def move(self, current: int, min_step: int,
                 max_step: int, goal: int) -> int:

            return 3
    # =====================================================

    ...

    def make_player(generic_name: str) -> Player:
        ...

```

```
        # =============== SOLUTION (Task 8.2) ================
        player_type_list = ['r', 'u', 's']

        while True:
            player_type = input(
                'Enter player type '
                '(r - Random Player, u - User Player, s - Strategic Player
    )')

            if player_type not in player_type_list:
                print('Please select one of the three values '
                      '({})'.format(','.join(player_type_list)))
                continue

            break

        if player_type == 'u':
            return UserPlayer(name)
        elif player_type == 's':
            return StrategicPlayer(name)
        elif player_type == 'r':
            return RandomPlayer(name)

        # ======================================================

    ...

    if __name__ == '__main__':
        # Uncomment the lines below to check your work using
        # python_ta and doctest.
        # import python_ta
        # python_ta.check_all(config={
        #     'extra-imports': ['random'],
        #     'allowed-io': [
        #         'main',
        #         'make_player',
        #         'move',
        #         'play_one_turn'
        #     ]
        # })
        main()
```

8.3) Tracking and reporting a player's record