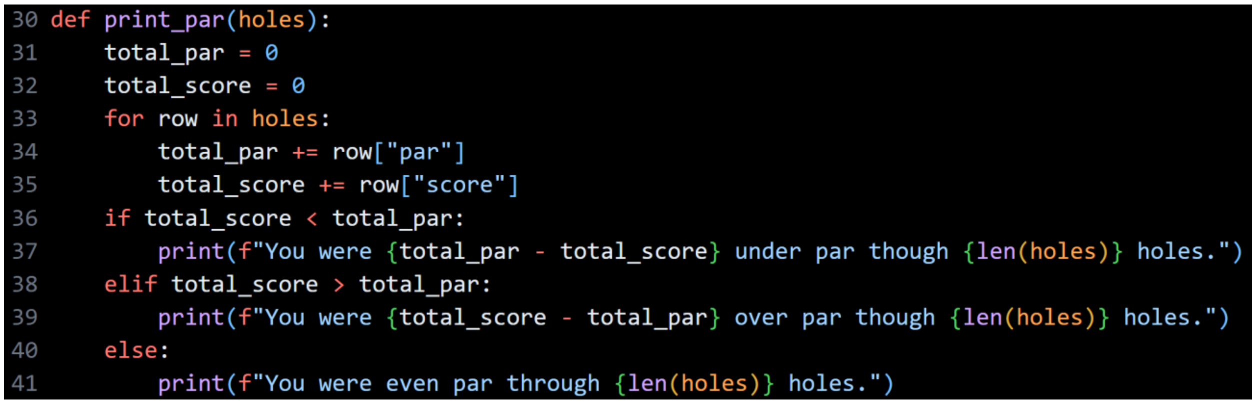
Procedure: Capture and paste two program code segments you developed during the administration of this task that contain a student-developed procedure that implements an algorithm used in your program and a call to that procedure.

i. The first program code segment must be a student-developed procedure that:

☐ Defines the procedure’s name and return type (if necessary)

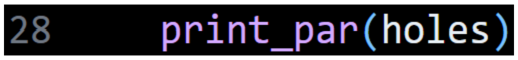
☐ Contains and uses one or more parameters that have an effect on the functionality of the procedure

☐ Implements an algorithm that includes sequencing, selection, and Iteration



```python
def print_par(holes):
    total_par = 0
    total_score = 0
    for row in holes:
        total_par += row["par"]
        total_score += row["score"]
    if total_score < total_par:
        print(f"You were {total_par - total_score} under par though {len(holes)} holes.")
    elif total_score > total_par:
        print(f"You were {total_score - total_par} over par though {len(holes)} holes.")
    else:
        print(f"You were even par through {len(holes)} holes.")
```

ii. The second program code segment must show where your student-developed procedure is being called in your program.

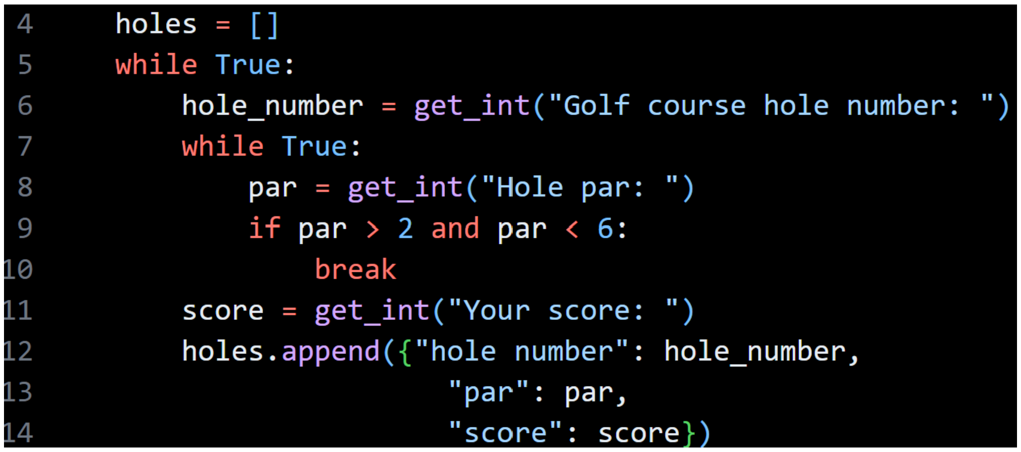


```python
    print_par(holes)
```

Mr. Price’s Personalized Project Reference

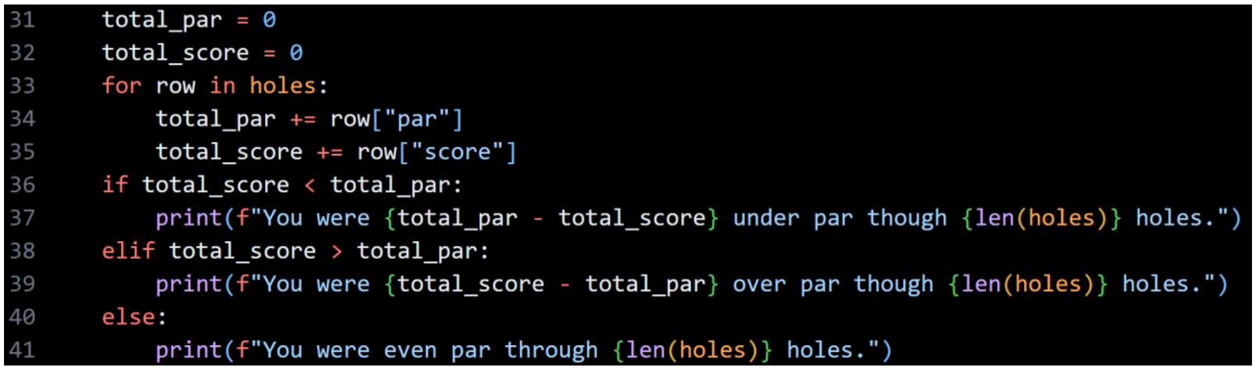
List: Capture and paste two program code segments you developed during the administration of this task that contain a list (or other collection type) being used to manage complexity in your program.

i. The first program code segment must show how data have been stored in the list.



```python
holes = []
while True:
    hole_number = get_int("Golf course hole number: ")
    while True:
        par = get_int("Hole par: ")
        if par > 2 and par < 6:
            break
    score = get_int("Your score: ")
    holes.append({"hole number": hole_number,
                  "par": par,
                  "score": score})
```

ii. The second program code segment must show the data in the same list being used, such as creating new data from the existing data or accessing multiple elements in the list, as part of fulfilling the program’s purpose.



```python
total_par = 0
total_score = 0
for row in holes:
    total_par += row["par"]
    total_score += row["score"]
if total_score < total_par:
    print(f"You were {total_par - total_score} under par though {len(holes)} holes.")
elif total_score > total_par:
    print(f"You were {total_score - total_par} over par though {len(holes)} holes.")
else:
    print(f"You were even par through {len(holes)} holes.")
```