Буду писать мысли по ходу появления.
жирным выделены названия файлов и номера строк в них

в общем проект произвел положительное впечатление:

- для сущностей есть классы,
- данные (тексты) отделены,
- есть общие функции для обработки ввода пользователя, декоратор для print,

Форматирование кода хорошее: автоформат в pycharm (ctrl+alt+L) добавил только дополнительные строки отступа в нескольких местах и пробелы перед \ и отсортировал импорты -- это мелочи. (но лучше все же автоформат использовать)

я бы использовал для форматирования f строки, а не конкатенацию с явным приведением к строке. Код будет компактнее и работать это будет быстрее и кушать меньше памяти. (можно и format использовать для сохранения совместимости со старыми версиями python). пример такого: classes.py:43

add_function.py
на мой взгляд название файла стоит поменять на более отражающее содержимое. Например, первое что пришло в голову банальное helpers.
6 - здорово что поддерживаются разные системы,
В названии лучше уточнить что именно очищается. например взять то что написано в доке к функции, получится более понятное имя)
наверно дальше не буду особо рассматривать именование, просто напишу идеи в этом направлении.

- когда пишем код представляем, что его будет читать человек, умеющий программировать, но не находящийся в контексте задачи, например мы сами через полгода. (это не только с именованию относится). из этих соображений и пытаемся сделать код читаемее
- если кажется, что стоит написать комментарий о том ЧТО делается, скорее всего стоит или переименовать что-то, или выделить в функцию или класс и тд. Тогда необходимость в комментарии может и пропасть (хоть и не всегда).

  Но безусловно нужно комментировать или писать доки для сложных регулярок, функций и прочих мест со сложной логикой, чтоб не нужно было смотреть что там внутри делается. Однако чтоб прочитать доку, надо прийти в объявление функции, а еще доку нужно поддерживать, поэтому хорошее имя очень помогает.
- Хорошо писать комментарии, чтобы пояснить, ЗАЧЕМ что-то делается.

53 - `or 12 <= d <= 14` это никогда не выполнится, тк покрыто в первом if, но и к ошибке не приведет по той же причине =)
Прикольная, кстати идея, поддержать корректный вывод слова, это делает текст более натуральным.
Я бы заленился это делать и выводил бы что-то типа “Количество монет в наличии: 10”, не так красиво, но зато только число подставить нужно.

если докопаться, то можно избавиться от первого условия, изменив второе и третье, тк в первом ничего не меняется

classes.py
хорошо что классы лежат отдельно. в случае если классы большие (не как тут) стоит каждый класс класть в отдельный файл

11 - здорово, что используется проперти, но для его названия нужно использовать существительное, а не глагол как для метода, тк мы к нему обращаемся как к атрибуту объекта
45 - В данном случает тут просится dataclass.
я размышлял о том, что здесь будет лучше - использовать датакласс, или перенести рандомайзную инициализацию атрибутов в init класса. Пришел к выводу, что при втором подходе будет увеличена связность объекта пирата с островом, поэтому наверно лучше текущий подход с инициализацией снаружи + датакласс =)

interactions.py
24 - можно избавиться от if elif с помощью словаря: https://youtu.be/MrfiwKZgpgo?t=461

71 - я не сразу понял почему ответов на 1 больше, может нужен комментарий зачем так
72 хотел написать, что лучше итерироваться по списку, а не по индексам и использовать enumerate
https://realpython.com/python-enumerate/#using-pythons-enumerate
но потому увидел что ты удаляешь из списка пиратов, тогда так итерироваться нельзя. не могу сказать, что мне нравится это итерирование по индексам, но сходу у меня не получилось придумать достойную альтернативу без значительного усложнения.
105 на мой взгляд стоит делать проверку внутри и возвращать bool, ну и назвать функцию соответствующим образом типа is_team_skilled_enough

main.py

16 функция main здоровая вышла, ее бы разбить на несколько, а может лучше сделать класс игры, чтоб было удобнее хранить данные и доставить их из методов, а то если просто разбить, придется передавать кучу параметров.
19, 20 - лучше вынести в константы

readme.md
Очень хорошо, что есть ридми, и что он на английском, тк это сразу расширяет аудиторию, которая сможет его прочитать.
Однако есть нюанс, который не сразу пришел мне в голову, тк мне не важно, на каком языке читать. Игра на русском языке и голосовать за нее должны быть русскоязычные участники группы, у большинства из которых, к сожалению, большие проблемы с английским, поэтому ридми еще и на русском потенциально мог дать дополнительное преимущество. обычно пишут что-то типа The English version is below

setup.py
хорошо что есть сетап, значит можно поставить игру как пакет, но чтобы было совсем клево, стоит сделать файл __main__ .py и запускать игру из него, тогда можно будет игру вызывать как python -m pgame https://habr.com/ru/post/456214/

texts.py
все константы надо сделать upper case