



# Python Programming

## (355)

**REGIONAL 2025**

APPLICATION KNOWLEDGE:

*TOTAL POINTS*

\_\_\_\_\_ (500 points)

**Test Time: 90 minutes**

**GENERAL GUIDELINES:**

*Failure to adhere to any of the following rules will result in disqualification:*

1. Contestant must hand in this test booklet and all printouts if any. Failure to do so will result in disqualification.
2. No equipment, supplies, or materials other than those specified for this event are allowed in the testing area. No previous BPA tests and/or sample tests (handwritten, photocopied, or keyed) are allowed in the testing area.
3. Electronic devices will be monitored according to ACT standards.

## Dungeon Master's Forge: Crafting Epic Adventures

In the realm of game development, creating an immersive, interactive experience is the ultimate goal. This problem invites you to step into the role of a game developer, crafting a Python-based text adventure game that simulates the classic dungeon crawl experience. Your task is to construct a virtual world where players can create a character, choose a weapon, and navigate through a series of perilous encounters within a dark and treacherous dungeon.

In this adventure game, players will face monstrous challenges, make strategic decisions, and strive to survive through three challenging levels. Each decision can lead to gold and health potions or to dire consequences, affecting the player's survival and success in uncovering the dungeon's hidden treasures.

### Contest Requirements:

You will have ninety (90) minutes to complete your work.

Your name and/or school name should *not* appear on work you submit for grading.

The provided student file, *RegionalPython.py*, is the file in which you will write your solution.

1. Create a folder using your *contestant number* as the name of the folder.
2. Copy your solution file into this folder. Your file *must* be a Python file (a .py file).
3. If submitting your test with a flash drive, copy your folder to the flash drive.
4. If submitting your test online, zip your folder and upload the zipped folder to the test submission website.
5. You will need to use a local Python IDE to complete this exam. No online interpreters for Python are allowed.
6. The graders will *not* compile or alter your source code if it does not run as-is.
7. Submissions that do *not* contain source code and/or files other than Python files will *not* be graded.

### What you will be doing:

1. **Include your contestant number as a comment at the top of the main source code file.**
2. Implement essential game functions: *display\_character*, *encounter*, *found\_loot*, *start\_game*, and *game\_over*.
3. Initiate the game by collecting player inputs for character creation (provided for you) and setting the scene for the dungeon adventure.
4. Design the encounter mechanics where players can choose to confront the monster or evade, influencing their journey and potential rewards.
5. Ensure the game progresses logically through three encounters, with the player's attributes and outcomes displayed after each encounter.
6. Conclude the game with a summary of the player's achievements and the treasure discovered, contingent on their survival and choices throughout the game.

### Development Standards:

1. Your Code must use a consistent variable naming convention.
2. All functions must be documented with comments explaining the purpose of the method, the input parameters (if any), and the output (if any). Readability is a trademark of good code.

**Commenting for Source Code Review (see the rubric):**

1. Certain sections of your code will be graded. These gradable blocks of code can range from creating data structures, method algorithms, exception handling, and class construction.
2. The grading rubric contains a section called Source Code Review: in this section is a list of descriptions of all the graded programming concepts.
3. Each gradable item must have a comment placed at its beginning, and the comment must be prefixed with the comment flag. The flag helps the graders easily locate the code to increase the effectiveness of grading.
4. The flag will *must* use the naming convention **SC#** (NOTE: the # symbol will be replaced with sequential numbering such as **SC1**, **SC2**, **SC3**, etc.)
5. Unless stated otherwise, NO explanation in the comment with the flag is required, only the comment flag; however, any information placed in the comment could help the grader better understand and avoid any costly errors.
6. The comment flag needs to be placed near the block of code it represents.
7. **If a comment flag is not present, you will not receive credit.**
8. Here is an example: the Source Code Review has a gradable section of code for printing to the console (this is just a generic example, not on the actual rubric):
  - Rubric:
    - SC12: *print* method in the *main* class is printing the correct object \_\_\_\_10 pts
  - Code snippet:
 

```
#SC12 printing the car object
print(car)
```

**Function Descriptions:**

- A. Function *display\_character* should provide a printout of a summary of the character in the following format (note that the Gold displays the number as two digits):

Name: Test	Level: 1
Gold: 00	Weapon: Crossbow
Health: 70	Strength: 17

- B. Function *found\_loot* should randomly choose to add to the player's gold or health. There should be a 70% chance that gold is chosen and a 30% chance that health is chosen.
- a. If gold is chosen, add a random amount to the gold -- between 25 and 150 -- and return the following:

```
__ gold!
```

The underscore should be the amount of gold found.

- b. If health is chosen, add a random amount to the health -- either 10, 20, or 30 -- and return the following:

```
a health potion. You restored __ health
```

The underscore should be the amount of health restored.

- C. Function ***start\_game*** should do the following, in addition to what is already provided:
- Set the global variables *name* and *weapon* to the appropriate user input that is provided in this function.
  - Display the following:

```
Hello, ____!
In this dungeon, you will fight three monsters.
If you survive to the end, treasure awaits!
You have your trusty _____, I see.
Good. You will need it.
Press Enter when you are ready to begin...
```

The first underscore should display the user's name and the second should display the user's weapon. You should also pause the program here until the user presses the Enter key.

- Use a for-loop to run the *encounter* function three times.
  - If, after any encounter, the user's health drops to zero or the user selected the option to run away, the loop should end.
  - Otherwise, increment the user's level by 1 and display the character summary.
- Once the loop of encounters is done, run the *game\_over* function.

D. Function ***encounter*** should go through the following logic:

- Display: "A monster is attacking you!"
- Ask the user if they would like to use their weapon or run away. Use the following format:

```
Enter:  '1' to use your Crossbow,
        '2' to run away
Choice: |
```

Note that the prompt mentions the user's specified weapon. The black line after "Choice: " is where the user's response should go. The user should only be allowed to enter either a '1' or a '2'. Any other value should return the phrase "Invalid choice!" and repeat the question. You must use a while-loop as well as a try/catch to make sure a valid option is entered.

- When the user enters a '2', nothing should happen in the function. However, when the user enters a '1':
  - Make a monster strength a random number from 10 to 20.
  - If the user's strength matches the monster's, or beats it, the user has won the fight. The *found\_loot* function should be called and you should display the following:

```
You defeated the monster and found _____!
Press Enter to continue
```

The underscore should show the result from the *found\_loot* function. You also need to pause the code here until the user presses the Enter key.

- If the user does not win the fight, you need to subtract 10 times the difference in strengths from the user's health and display the following:

```
That was rough! you lost ____ health.
```

The underscore should show the amount of health lost. If this were to drop the user's health to below zero, set the health to zero instead. If there is still health left, display the following:

```
Luckily you managed to get past the monster!  
Press Enter to continue
```

Again, the code should pause here until the user presses the Enter key.

- d. This function should return the user's choice.

E. Function ***game\_over*** should determine how the game ends:

- a. If the user still has health left and has reached level 4, you should add a random amount to the gold (between 500 and 5000) and display the following:

```
You made it to the treasure! You found ____ gold!
```

The underscore should show the amount of gold found.

- b. If the user has health left but has not reached level 4, you should display the following:

```
You didn't find the treasure, but you survived to fight again another day...
```

- c. If the user has lost all their health, display the following:

```
You fought as best you could, but didn't make it.  
The treasure waits for the next adventurer...
```

- d. No matter what ending the user earned, finish this function by displaying the character summary.

Here are several example runs of a completed program, each example illustrating one of the possible endings.

### Sample Run #1: Player Loses

```
Welcome to the dungeon!
What is your name, adventurer? Sample 1
What is your weapon of choice? Sword

Name: Sample 1      Level: 1
Gold: 00            Weapon: Sword
Health: 70          Strength: 11

Hello, Sample 1!
In this dungeon, you will fight three monsters.
If you survive to the end, treasure awaits!
You have your trusty Sword, I see.
Good. You will need it.
Press Enter when you are ready to begin...

A monster is attacking you!
Enter: '1' to use your Sword,
      '2' to run away
Choice: 1

That was rough! you lost 40 health.
Luckily you managed to get past the monster!
Press Enter to continue

Name: Sample 1      Level: 2
Gold: 00            Weapon: Sword
Health: 30          Strength: 11

A monster is attacking you!
Enter: '1' to use your Sword,
      '2' to run away
Choice: 1

That was rough! you lost 10 health.
Luckily you managed to get past the monster!
Press Enter to continue

Name: Sample 1      Level: 3
Gold: 00            Weapon: Sword
Health: 20          Strength: 11

A monster is attacking you!
Enter: '1' to use your Sword,
      '2' to run away
Choice: 1

That was rough! you lost 40 health.

You fought as best you could, but didn't make it.
The treasure waits for the next adventurer...

Name: Sample 1      Level: 3
Gold: 00            Weapon: Sword
Health: 0           Strength: 11
```

### Sample Run #3: Player Quits

```
Welcome to the dungeon!
What is your name, adventurer? Sample 3
What is your weapon of choice? Dagger

Name: Sample 3      Level: 1
Gold: 00            Weapon: Dagger
Health: 100         Strength: 14

Hello, Sample 3!
In this dungeon, you will fight three monsters.
If you survive to the end, treasure awaits!
You have your trusty Dagger, I see.
Good. You will need it.
Press Enter when you are ready to begin...

A monster is attacking you!
Enter: '1' to use your Dagger,
      '2' to run away
Choice: 1

That was rough! you lost 40 health.
Luckily you managed to get past the monster!
Press Enter to continue

Name: Sample 3      Level: 2
Gold: 00            Weapon: Dagger
Health: 60          Strength: 14

A monster is attacking you!
Enter: '1' to use your Dagger,
      '2' to run away
Choice: 1

You defeated the monster and found 29 gold!
Press Enter to continue

Name: Sample 3      Level: 3
Gold: 29            Weapon: Dagger
Health: 60          Strength: 14

A monster is attacking you!
Enter: '1' to use your Dagger,
      '2' to run away
Choice: 2

You didn't find the treasure, but you survived to fight again another day...

Name: Sample 3      Level: 3
Gold: 29            Weapon: Dagger
Health: 60          Strength: 14
```

## Sample Run #2: Player Wins

```
Welcome to the dungeon!
What is your name, adventurer? Sample 2
What is your weapon of choice? Axe

Name: Sample 2      Level: 1
Gold: 00            Weapon: Axe
Health: 70          Strength: 19

Hello, Sample 2!
In this dungeon, you will fight three monsters.
If you survive to the end, treasure awaits!
You have your trusty Axe, I see.
Good. You will need it.
Press Enter when you are ready to begin...

A monster is attacking you!
Enter: '1' to use your Axe,
      '2' to run away
Choice: 1

You defeated the monster and found 54 gold!
Press Enter to continue

Name: Sample 2      Level: 2
Gold: 54            Weapon: Axe
Health: 70          Strength: 19

A monster is attacking you!
Enter: '1' to use your Axe,
      '2' to run away
Choice: 1

You defeated the monster and found a health potion. You restored 30 health!
Press Enter to continue

Name: Sample 2      Level: 3
Gold: 54            Weapon: Axe
Health: 100         Strength: 19

A monster is attacking you!
Enter: '1' to use your Axe,
      '2' to run away
Choice: 1

You defeated the monster and found 94 gold!
Press Enter to continue

Name: Sample 2      Level: 4
Gold: 148           Weapon: Axe
Health: 100         Strength: 19

You made it to the treasure! You found 2602 gold!

Name: Sample 2      Level: 4
Gold: 2750          Weapon: Axe
Health: 100         Strength: 19
```

Your application will be graded on the following criteria:

### **Solution and Project**

The project is present on the flash drive / uploaded as a zipped folder. \_\_\_\_\_20 pts

The required file is in one folder; the folder name is your contestant ID \_\_\_\_\_10 pts

### **Program Execution**

Code copied to USB drive and the program runs from USB \_\_\_\_\_30 pts

*If the program does not execute, then the remaining items in this section receive a score of zero.*

Game start: after user enters a name and weapon, the introductory message displayed should incorporate the user responses in it ("Hello, \_\_\_\_", "your trusty \_\_\_\_"). \_\_\_\_\_20 pts

Program pauses at the phrases "Press Enter when you are ready to begin..." and "Press Enter to continue" and moves on when the user presses the enter key. \_\_\_\_\_10 pts

Program displays the message "A monster is attacking you!", followed by two options: use the weapon or run away. The weapon option should display the specific weapon entered by the user. \_\_\_\_\_10 pts

Program should accept only a '1' or '2' for the input. Any other response should display the message "Invalid choice!" and repeat the prompt. \_\_\_\_\_30 pts

If '1' is entered, program should display either "You defeated the monster and found \_\_\_\_!" followed by "Press Enter to continue" or "That was rough! you lost \_\_\_\_health.". \_\_\_\_\_10 pts

If program displays "You defeated the monster and found \_\_\_\_!", the underscore should be replaced with either "\_\_\_\_gold" or "a health potion. You restored \_\_\_\_health". These underscores should be replaced with actual numbers. \_\_\_\_\_20 pts

If program displays "That was rough! you lost \_\_\_\_health.", the underscore should be replaced with a number. \_\_\_\_\_20 pts

After the result of the encounter, the character summary is displayed. \_\_\_\_\_10 pts

Program should go through three encounters, unless all the user's health is lost or the user enters a '2'. \_\_\_\_\_20 pts

If all the health is lost before the end of the third encounter, program displays "You fought as best you could, but didn't make it." followed by "The treasure waits for the next adventurer...". \_\_\_\_\_20 pts

If the user enters a '2', program should display "You didn't find the treasure, but you survived to fight again another day...". \_\_\_\_\_20 pts

If all three encounters resolve and there is still health left, program should display "You made it to the treasure! You found \_\_\_\_gold!". The underscore should be replaced with a number. \_\_\_\_\_20 pts

**Subtotal** \_\_\_\_\_/270

**Source Code Review**

Contestant number is commented at the top of the Python file. \_\_\_\_\_ 10 pts

*NOTE: you **must** place the comment flag in front of the comment in your code **to get credit**. The comment flag will precede the explanation. For example, if the flag is SC1, your comment must read as “#SC1...” in front of the part of the code being reviewed. Code must work to get credit.*

SC1: Function **display\_character** prints the name, level, gold, weapon, health, and strength variables in the format specified. \_\_\_\_\_ 20 pts

SC2: Function **display\_character** formats the gold variable to a two digit number. \_\_\_\_\_ 10 pts

SC3: Function **found\_loot** randomly chooses between gold and health, with a 70% chance to choose gold and a 30% chance to choose health. \_\_\_\_\_ 30 pts

SC4: Function **found\_loot** adds a random amount of gold (between 25 and 150) to gold variable or health (10, 20, or 30) to the health variable, depending on which gets chosen. \_\_\_\_\_ 20 pts

SC5: Function **start\_game** uses a for-loop to run the *encounter* function three times. \_\_\_\_\_ 10 pts

SC6: Function **start\_game** either ends the loop if the health variable hits zero or if the user input from the *encounter* function is a ‘2’, or adds one to the level variable if the loop doesn’t end. \_\_\_\_\_ 20 pts

SC7: Function **start\_game** makes a call to *game\_over* after the loop \_\_\_\_\_ 10 pts

SC8: Function **encounter** uses a while loop and conditionals to ensure that the only acceptable user input is a ‘1’ or a ‘2’. \_\_\_\_\_ 20 pts

SC9: Function **encounter** assigns the monster strength a random number between 10 and 20. If the monster’s strength is higher, subtract 10 times the difference between the monster strength and the user strength from the user health. Or, if the user’s strength is higher, make a call to *found\_loot*. \_\_\_\_\_ 30 pts

SC10: Function **encounter** returns the user input. \_\_\_\_\_ 10 pts

SC11: Function **game\_over** uses conditionals to establish three possible scenarios based on the health and level variables, as described in the instructions. \_\_\_\_\_ 20 pts

SC12: Function **game\_over** adds a random number between 500 and 5000 to the gold variable. \_\_\_\_\_ 10 pts

SC13: Global variables are accessed and used throughout all four methods. \_\_\_\_\_ 10 pts

**Subtotal** \_\_\_\_\_/230

**Total Points** \_\_\_\_\_/500

## Completed Source Code

```
import random

# Starting attributes
name = ""
weapon = ""
health = random.randint(7,10) * 10
gold = 0
strength = random.randint(12,17)
level = 1

# Character Display
def display_character():
    print("\nName: %s\tLevel: %s" % (name, level))
    print("Gold: %02d\tWeapon: %s" % (gold, weapon))
    print("Health: %d\tStrength: %d\n\n" % (health, strength))

# Monster encounter
def encounter():
    print("A monster is attacking you!")
    while True:
        try:
            attack = int(input("Enter:\t'1' to use your %s,\n\t'2' to run away\nChoice: " % (weapon)))
            if attack < 1 or attack > 2:
                print("Invalid choice!")
            else:
                break
        except ValueError:
            print("Invalid Choice!")
    if attack == 1:
        monster = random.randint(10,20)
        result = strength - monster
        if(result >= 0):
            reward = found_loot()
            input("\nYou defeated the monster and found %s!\nPress Enter to continue" % (reward))
        else:
            global health
            health -= abs(result) * 10
            print("\nThat was rough! you lost %d health." % (abs(result) * 10))
            if(health <= 0):
                health = 0
            else:
                input("Luckily you managed to get past the monster!\nPress Enter to continue")
    return attack

# Reward for winning an encounter
def found_loot():
    global gold, health
    loot = random.choice('ggggggghhh')
    if loot == 'g':
```

```

        new_gold = random.randint(25,150)
        gold += new_gold
        return "%d gold" % (new_gold)
    else:
        bonus = random.randint(1,3) * 10
        health += bonus
        return "a health potion. You restored %d health" % (bonus)

# Start game
def start_game():
    global level, health, name, weapon
    print("Welcome to the dungeon!")
    enter_name = input("What is your name, adventurer? ")
    enter_weapon = input("What is your weapon of choice? ")
    name = enter_name
    weapon = enter_weapon
    display_character()
    print("\nHello, %s!\nIn this dungeon, you will fight three monsters.\nIf you survive to the end, treasure awaits!\nYou
    have your trusty %s, I see.\nGood. You will need it." % (enter_name, enter_weapon))
    input("Press Enter when you are ready to begin...")
    for i in range(3):
        result = encounter()
        if health <= 0 or result == 2:
            break
        else:
            level += 1
            display_character()
    game_over()

# End game
def game_over():
    global health, level, gold
    if health > 0 and level == 4:
        treasure = random.randint(500,5000)
        gold += treasure
        print("\nYou made it to the treasure! You found %d gold!" % (treasure))
    elif health > 0 and level < 4:
        print("\nYou didn't find the treasure, but you survived to fight again another day...")
    else:
        print("\nYou fought as best you could, but didn't make it.\nThe treasure waits for the next adventurer...")
    display_character()

# Main code
start_game()

```