← Back to Login

Student: Aayush.shrestha1@westcliff.edu

🖨 Print Report

Exam: PYTHON

Completed: 9/18/2025, 4:10:46 AM

Your Score	Correct	Incorrect
91.38% 53/58 points	39 out of 46	7 questions missed

Status

Expired

Violations: 3

⚠️ Security Violations Detected

The following security violations were detected during your exam:

Contextmenu	1	Window Blur	2

📝 Question-by-Question Review

1 ✓ Correct mcq_single 1/1 points

Which symbol is used for comments in Python?

Options:

○ //

○ # ✓ Correct

○ /*

○ --

2 ✓ Correct mcq_single 1/1 points

What is the correct way to create a variable in Python?

Options:

○ var x = 5

○ int x = 5

○ x = 5 ✓ Correct

○ declare x = 5

3 ✓ Correct mcq_single 1/1 points

Which data type is used to store text in Python?

Options:

○ int

○ float

○ str ✓ Correct

○ char

4 ✓ Correct mcq_single 2/2 points

What is the output of print(type(5))?

Options:

○ <class 'int'> ✓ Correct

○ <class 'float'>

○ <class 'str'>

○ int

5 ✕ Incorrect mcq_single 0/1 points

Which operator is used for exponentiation in Python?

Options:

○ ^

○ ** ✓ Correct

○ exp ✕ Your Answer

○ pow

6 ✓ Correct mcq_single 1/1 points

What is the correct way to check if two values are equal?

Options:

○ x = y

○ x == y ✓ Correct

○ x := y

○ x equals y

7 ✓ Correct mcq_single 1/1 points

Which function is used to get input from the user?

Options:

○ input() ✓ Correct

○ get()

○ read()

○ scan()

8 ✓ Correct mcq_single 1/1 points

What is the output of len('Hello')?

Options:

○ 4

○ 5 ✓ Correct

○ 6

○ Error

9 ✓ Correct mcq_single 2/2 points

Which of the following is a valid variable name in Python?

Options:

○ 2var

○ var-2

○ _var2 ✓ Correct

○ var 2

10 ✓ Correct mcq_single 2/2 points

What is the result of bool('')?

Options:

○ True

○ False ✓ Correct

○ None

○ Error

11 ✓ Correct mcq_single 1/1 points

Which method converts a string to uppercase?

Options:

○ upper() ✓ Correct

○ uppercase()

○ toUpper()

○ capitalize()

12 ✓ Correct mcq_single 2/2 points

What is the output of str(123)?

Options:

○ 123

○ '123' ✓ Correct

○ Error

○ None

13 ✓ Correct mcq_single 1/1 points

How do you create an empty list in Python?

Options:

○ list = {}

○ list = [] ✓ Correct

○ list = ()

○ list = ''

14 ✓ Correct mcq_single 1/1 points

Which method adds an element to the end of a list?

Options:

○ add()

○ append() ✓ Correct

○ insert()

○ push()

15 ✓ Correct mcq_single 2/2 points

What is the output of [1, 2, 3][1]?

Options:

○ 1

○ 2 ✓ Correct

○ 3

○ Error

16 ✓ Correct mcq_single 1/1 points

How do you get the length of a list named 'my_list'?

Options:

○ my_list.length

○ len(my_list) ✓ Correct

○ my_list.size()

○ count(my_list)

17 ✓ Correct mcq_single 1/1 points

Which keyword is used to start a conditional statement?

Options:

○ if ✓ Correct

○ when

○ check

○ condition

18 ✕ Incorrect mcq_single 0/1 points

What comes after 'if' when you want to check another condition?

Options:

○ else

○ elif ✓ Correct

○ elseif

○ then

19 ✓ Correct mcq_single 1/1 points

Which keyword is used to create a for loop?

Options:

○ for ✓ Correct

○ loop

○ repeat

○ iterate

20 ✓ Correct mcq_single 2/2 points

What does range(3) produce?

Options:

○ [1, 2, 3]

○ [0, 1, 2] ✓ Correct

○ [0, 1, 2, 3]

○ 3

21 ✕ Incorrect mcq_single 0/1 points

Which statement is used to exit a loop early?

Options:

○ break ✓ Correct

○ exit ✕ Your Answer

○ stop

○ end

22 ✓ Correct mcq_single 1/1 points

Which keyword is used to define a function in Python?

Options:

○ function

○ def ✓ Correct

○ func

○ define

23 ✓ Correct mcq_single 1/1 points

What keyword is used to return a value from a function?

Options:

○ return ✓ Correct

○ output

○ give

○ send

24 ✓ Correct mcq_single 2/2 points

What is returned by a function that has no return statement?

Options:

○ 0

○ None ✓ Correct

○ False

○ Empty string

25 ✓ Correct mcq_single 1/1 points

How do you call a function named 'my_function' with no parameters?

Options:

○ my_function

○ my_function() ✓ Correct

○ call my_function

○ my_function[]

26 ✓ Correct mcq_single 1/1 points

Which function is used to open a file in Python?

Options:

○ open() ✓ Correct

○ file()

○ read()

○ load()

27 ✓ Correct mcq_single 1/1 points

Which mode is used to read a file?

Options:

○ 'w'

○ 'r' ✓ Correct

○ 'a'

○ 'x'

28 ✓ Correct mcq_single 2/2 points

Which statement is recommended for file handling?

Options:

○ open() and close()

○ with open() ✓ Correct

○ file() and end()

○ read() and write()

29 ✓ Correct mcq_single 1/1 points

Which keyword is used to create a class in Python?

Options:

○ class ✓ Correct

○ object

○ create

○ new

30 ✓ Correct mcq_single 2/2 points

What is the special method used to initialize an object?

Options:

○ __start__()

○ __init__() ✓ Correct

○ __new__()

○ __create__()

31 ✓ Correct mcq_single 2/2 points

What does 'self' refer to in a Python class?

Options:

○ The class itself

○ The current instance ✓ Correct

○ The parent class

○ A keyword

32 ✓ Correct mcq_single 1/1 points

Which keyword is used to handle exceptions in Python?

Options:

○ catch

○ except ✓ Correct

○ handle

○ error

33 ✓ Correct mcq_single 1/1 points

Which keyword is used with 'except' to handle exceptions?

Options:

○ try ✓ Correct

○ catch

○ handle

○ error

34 ✓ Correct mcq_single 2/2 points

What is Django?

Options:

○ A Python library for data science

○ A web framework for Python ✓ Correct

○ A database management system

○ A Python IDE

35 ✓ Correct mcq_single 2/2 points

Which command is used to create a new Django project?

Options:

○ django-admin startproject ✓ Correct

○ django create project

○ python manage.py startproject

○ django new project

36 ✓ Correct mcq_single 1/1 points

What is the default port for Django development server?

Options:

- ○ 8080
- ○ 3000
- ○ 8000 ✓ Correct
- ○ 5000

37 ✕ Incorrect mcq_single 0/2 points

Which file contains Django project settings?

Options:

- ○ config.py
- ○ settings.py ✓ Correct
- ○ django.py ✕ Your Answer
- ○ manage.py

38 ✓ Correct mcq_single 2/2 points

Which command is used to run the Django development server?

Options:

- ○ python manage.py run
- ○ python manage.py start
- ○ python manage.py runserver ✓ Correct
- ○ django runserver

39 ✓ Correct mcq_single 2/2 points

What does MVT stand for in Django architecture?

Options:

○ Model View Template ✓ Correct

○ Model View Type

○ Module View Template

○ Model Variable Template

40 ✓ Correct mcq_single 1/1 points

Which module is used for mathematical operations in Python?

Options:

○ math ✓ Correct

○ calculate

○ numbers

○ arithmetic

41 ✓ Correct mcq_single 1/1 points

Which module is used for random number generation?

Options:

○ random ✓ Correct

○ rand

○ number

○ generate

42 ✓ Correct mcq_single 1/1 points

Which module is used for date and time operations?

Options:

○ time

○ datetime ✓ Correct

○ date

○ calendar

43 ✓ Correct mcq_single 1/1 points

How do you import a module named 'math'?

Options:

○ include math

○ import math ✓ Correct

○ use math

○ require math

44 ✕ Incorrect code 0/0 points

🏦 Bank Account Class Implementation

Create a 'BankAccount' class with the following features:

📋 Requirements:

• Constructor that takes account_number and initial_balance

• deposit(amount) method that adds money to balance

• withdraw(amount) method that subtracts money (only if sufficient balance)

• get_balance() method that returns current balance

🧪 Testing:

Create an account, deposit 500, withdraw 200, and print the final balance

Correct Answer: class BankAccount: def __init__(self, account_number, initial_balance): self.account_number = account_number self.balance = initial_balance def deposit(self, amount): self.balance += amount def withdraw(self, amount): if amount <= self.balance: self.balance -= amount else: print("Insufficient balance") def get_balance(self): return self.balance # Test the class account = BankAccount("123456", 1000) account.deposit(500) account.withdraw(200) print(f"Final balance: {account.get_balance()}")

Your Answer: class BankAccount: def __init__(self, account_number, initial_balance): self.account_number = account_number self.balance = initial_balance def deposit(self, amount): self.balance += amount print (f"Deposited: $ {amount}. New balance: $ {self.balance}") def withdraw(self, amount): if amount <= self.balance: self.balance -= amount print(f"Withdrew: $ {amount}. New balance: $ {self.balance}") else: print("Insufficient funds!") def get_balance(self): return self.balance #Testing account = BankAccount("123456", 0) #Deposit account.deposit(500) #Withdraw account.withdraw(200) #Print final balance print(f"Final balance: $ {account.get_balance()}")

45 ✕ Incorrect code 0/0 points

📚 Library Management System

Create a 'Library' class that manages books:

📋 Class Requirements:

• Constructor initializes an empty list of books

• add_book(title, author) method adds a book as a dictionary

• remove_book(title) method removes a book by title

• find_books_by_author(author) method returns all books by that author

• display_all_books() method prints all books in format: "Title by Author"

🧪 Testing Instructions:

Test by adding 3 books, removing 1, and finding books by a specific author

💡 Tip: Use list comprehension for filtering operations

Correct Answer: class Library: def __init__(self): self.books = [] def add_book(self, title, author): book = {'title': title, 'author': author} self.books.append(book) def remove_book(self, title): self.books = [book for book in self.books if book['title'] != title] def find_books_by_author(self, author): return [book for book in self.books if book['author'] == author] def display_all_books(self): for book in self.books: print(f"{book['title']} by {book['author']}") # Test the library library = Library() library.add_book("1984", "George Orwell") library.add_book("Animal Farm", "George Orwell") library.add_book("To Kill a Mockingbird", "Harper Lee") library.remove_book("Animal Farm") orwell_books = library.find_books_by_author("George Orwell") print("Books by George Orwell:", orwell_books) library.display_all_books()

Your Answer: class Library: def __init__(self): self.books = [] def add_book(self, title, author): book = {"title": title, "author": author} self.books.append(book) print(f"Added: {title} by {author}") def remove_book(self, title): for book in self.books: if book["title"] == title: self.books.remove(book) print(f"Removed: {title}") return print(f"Book '{title}' not found") def find_books_by_author(self, author): return [book for book in self.books if book["author"] == author] def display_all_books(self): if not self.books: print ("No books in library") return print("All books in library:") for book in self.books: print(f"{book['title']} by {}book['author']}") #Testing the library system library = Library() library.add_book("1920", "Salman Khan") library.add_book("Charlie and the Chocolate factory", "Irfan Khan") library.add_book("Mero jivan katha", "Salman Khan") # Displaying the books library.display_all_books() # Removing a book library.remove_book("Charlie and the Chocolate factory") # Display the books after removing one library.display_all_books() # Finding books by author Khan_books = library.find_books_by_author("Salman Khan") print(f"Books by Salman Khan: {len(orwell_books)}") for books in Khan_books: print(f"- {book['title']}")

46 ✕ Incorrect code 0/0 points

🧑‍🎓 Student Class with Inheritance

Create a 'Student' class with inheritance:

📋 Base Class Requirements:

🏷️ Person Class:

- Constructor: name, age
- Method: display_info() - prints name and age

📋 Derived Class Requirements:

🎓 Student Class (inherits from Person):

- Constructor: name, age, student_id
- Properties: grades list (initialized empty)
- Methods:

- add_grade(grade) - adds grade to list

- calculate_average() - returns average of all grades

- display_student_info() - shows all info including grades and average

🧪 Testing Instructions:

Create a student object, add some grades (85, 92, 78, 88), and display all information

💡 Tip: Use super() to call parent class constructor

Correct Answer: class Person: def __init__(self, name, age): self.name = name self.age = age def display_info(self): print(f"Name: {self.name}, Age: {self.age}") class Student(Person): def __init__(self, name, age, student_id): super().__init__(name, age) self.student_id = student_id self.grades = [] def add_grade(self, grade): self.grades.append(grade) def calculate_average(self): if self.grades: return sum(self.grades) /

len(self.grades) return 0 def display_student_info(self): self.display_info() print(f"Student ID: {self.student_id}") print(f"Grades: {self.grades}") print(f"Average: {self.calculate_average():.2f}") # Test the classes student = Student("John Doe", 20, "S12345") student.add_grade(85) student.add_grade(92) student.add_grade(78) student.add_grade(88) student.display_student_info()

Your Answer: class Person: def __init__(self, name, age): self.name = name self.age + age def display_info(self): print(f"Name: {self.name}, Age: {self.age}") class Student(Person): def __init__()self, name, age, student_id): super().__init__(name, age) self.student_id = student_id self.grades = [] def add_grades(self, grade): self.grades.append(grade) print(f"Added grade: {grade}") def calculate_average(self): if self.grades: return sum(self.grades) / len(self.grades) return 0 def display_student_info(self): print(f"Student ID: {self.student_id}") self.display_info() print(f"Grades: {self.grades")

Thank you for taking the exam. You can print this report for your records.

Return to Login