Quiz Details

Quiz Instructions:

Show Question Details

Reproducibility

Group Name Pick 3 questions, 1 pts per question Pick ? questions, ? pts per question

Cancel Update

OS 1 pts

Which of the following allocates and abstracts computing resources?

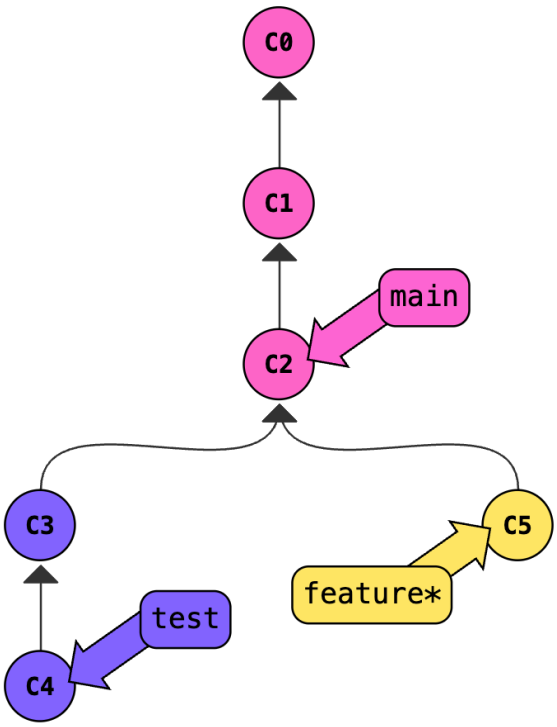
- operating system
- CPU
- interpreter
- instruction set

detached head 1 pts

Which of the following git commands will switch a repository to the "detached head" state?

- git checkout commit_number
- git checkout branch_name
- git checkout main
- git checkout -b branch_name

Branching & Merging 1 pts



The current state of my git repository is given by the above image. Which of the following git commands might cause a merge conflict?

- `git checkout main`
`git merge test`
- `git checkout main`
`git merge feature`
- `git checkout test`
`git merge feature`
- `git checkout feature`
`git merge main`

Performance

Group Name Pick 4 questions, 1 pts per question Pick ? questions, ? pts per question

Cancel Update

check_output 1 pts

Which of the following will correctly run `git diff` on the "development" directory using `check_output` ?

- `check_output("git diff", cwd="development", shell=True)`
- `check_output("git diff", cwd="development")`
- `check_output(["git", "diff"], cwd="development", shell=True)`
- `check_output(cwd="development", ["git", "diff"])`

Complexity 1 pts

```
sum = 0
for i in A[:100]:
    for j in B[-100:]:
        sum += i * j
```

Given A is a list of M elements and B is a list of N elements, what's the time complexity of the above code?

- O(1)
- O(MN)
- O(M)
- O(N)

Complexity List Operation 1 pts

Given L is a list of N elements, which of the following list operations have O(1) time complexity?

1. `len(L)`
2. `L.pop(10)`
3. `L[-10]`
4. `L.insert(-10, "10")`

- 1, 3 and 4

- 1, 2 and 4
- 3 and 4
- 2 and 4
- 1 and 3

Memory 1 pts

Which of the following allows me to save storage space when processing a large dataset stored in a zip?

- zipfile.ZipFile
- io.TextIOWrapper
- csv.DictReader

Recursion

Group Name Pick 2 questions, 1 pts per question Pick ? questions, ? pts per question

Cancel Update

Recursion Return 1 pts

```
def mystery(lst):
    if len(lst) <= 1:
        return 0
    return lst[1] + mystery(lst[2:])
```

What does `mystery([1, 4, 5, 3, 2, 1, 4])` return?

- 8
- 12
- 20
- 7

Recursion Print 1 pts

```
def foo(n):
    if n < 1:
        return
    else:
        foo(n-1)
        print(n)
        foo(n-1)
```

What does `foo(3)` print?

- 1 2 3 2 1
- 1 2 1 3 1 2 1
- 3 2 1 2 3
- 3 2 3 1 3 2 3

OOP

Group Name Pick 4 questions, 1 pts per question Pick ? questions, ? pts per question

Cancel Update

Special Methods 2 1 pts

```
settings = {"value": 0}

class CustomContext:
    def __init__(self, diff):
        self.diff = diff

    def __enter__(self):
        settings["value"] += 2 * self.diff

    def __exit__(self, exc_type, exc_value, traceback):
        pass

values = []
values.append(settings["value"])

with CustomContext(2):
    values.append(settings["value"])

    with CustomContext(1):
        values.append(settings["value"])

        with CustomContext(3):
            values.append(settings["value"])

        values.append(settings["value"])
    values.append(settings["value"])
values.append(settings["value"])

print(values)
```

What's printed in the last line of the above code?

- [0, 4, 6, 12, 12, 12, 12]
- [0, 0, 0, 0, 0, 0, 0]
- [0, 4, 6, 12, 9, 8, 6]
- [0, 4, 6, 12, 6, 4, 0]

Special Methods 1 1 pts

What of the following special methods defines what will be returned when we use `print()` on an object?

- __str__
- __repr__
- _repr_html_

Positional Arguments 1 pts

```
class Car:
    def __init__(self, brand, color, mode="oil"):
        self.brand = brand
        self.color = blue
        self.mode = mode

car = Car("Toyota", "blue")
```

How many positional arguments are passed to Car.__init__() when creating a Car object in the last line of the above code?

- 3
- 2
- 4
- 5

Inheritance 1 pts

```
class Calculator:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def sum(self):
        return self.x + self.y

    def subtract(self):
        return self.x - self.y
```

```
class AdvancedCalculator(Calculator):
    def __init__(self, x, y, z):
        super().__init__(x, y)
        self.z = z

    def sum(self):
        return self.x - self.z

    def magic(self):
        return self.sum() * self.subtract()

calc = AdvancedCalculator(1, 2, 3)
print(calc.magic())
```

What's printed on the last line of the above code?

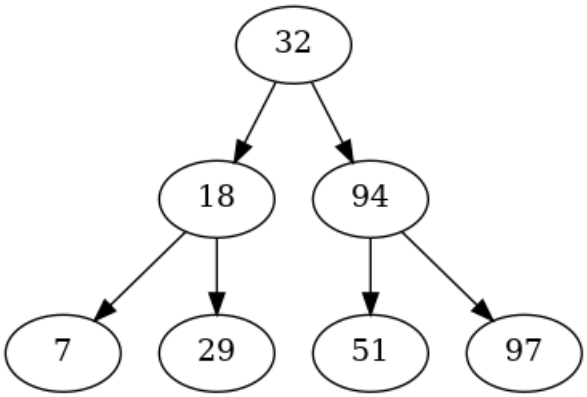
- 2
- -3
- 6
- -2

Graph

Group Name Pick 3 questions, 1 pts per question Pick ? questions, ? pts per question

Cancel Update

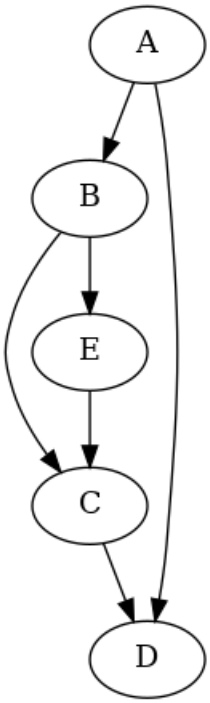
BST Insertion Order 1 pts



Consider the BST insertion algorithm we learned in class. Given the above BST, which of the following **CANNOT** be the insertion order?

- [32, 18, 7, 51, 94, 29, 97]
- [32, 18, 29, 94, 7, 51, 97]
- [32, 94, 51, 97, 18, 29, 7]
- [32, 18, 94, 29, 97, 51, 7]

Connectivity and Cycles 1 pts



Which of the following statements about the above graph is true?

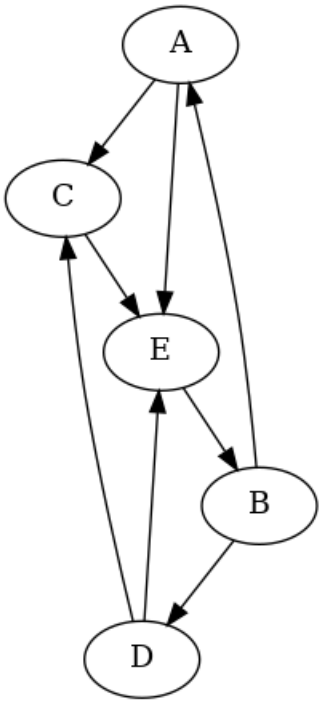
- The graph is acyclic and weakly connected.
- The graph is cyclic and weakly connected.
- The graph is cyclic and strongly connected.
- The graph is acyclic and strongly connected.

DAG 1 pts

Let's build a graph to represent inheritance. Each class is a node, and each parent class has edges pointing to its child classes. Which of the following **MOST SPECIFICALLY** describes this graph.

- DAG
- tree
- binary tree
- graph

DFS 1 pts



Assume the children list is in alphabetical order. Given the above graph, which of the following is the path from B to E that is returned from **depth first search**.

- (B, A, C, E)
- (B, E)
- (B, A, E)
- (B, D, E)
- (B, D, C, E)

BFS 1 pts

Assume the children list is in alphabetical order. Given the graph in the previous question, which of the following is the path from B to E that is returned from **breadth first search**.

- (B, A, E)
- (B, A, C, E)
- (B, E)
- (B, D, E)
- (B, D, C, E)

deque/heapq 1 pts

```
import heapq
from collections import deque

numbers = [18, 3, 7, 11, 2, 6]
q = deque()
heap = numbers
heapq.heapify(heap)
for i in range(len(numbers)):
    q.appendleft(heapq.heappop(heap))

print(q)
```

What's printed in the last line of the above code?

- deque([18, 11, 7, 6, 3, 2])
- deque([2, 3, 6, 7, 11, 18])
- deque([18, 3, 2, 6, 7, 11])

- deque([6, 2, 11, 7, 3, 18])

Heapq Complexity 1 pts

Let's replace `numbers` in the code snippet from the previous question with list `L` of N elements. What's the time complexity of the code snippet with the replaced list `L`?

- O(N * logN)
- O(N**2)
- O(N**2 * logN)
- O(N)