

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING																		
Program Name: B. Tech		Assignment Type: Lab	Academic Year: 2025-2026																	
Course Coordinator Name		Dr. Rishabh Mittal																		
Instructor(s) Name		<table border="1"> <tr><td>Mr. S Naresh Kumar</td></tr> <tr><td>Ms. B. Swathi</td></tr> <tr><td>Dr. Sasanko Shekhar Gantayat</td></tr> <tr><td>Mr. Md Sallauddin</td></tr> <tr><td>Dr. Mathivanan</td></tr> <tr><td>Mr. Y Srikanth</td></tr> <tr><td>Ms. N Shilpa</td></tr> <tr><td>Dr. Rishabh Mittal (Coordinator)</td></tr> <tr><td>Dr. R. Prashant Kumar</td></tr> <tr><td>Mr. Ankushavali MD</td></tr> <tr><td>Mr. B Viswanath</td></tr> <tr><td>Ms. Sujitha Reddy</td></tr> <tr><td>Ms. A. Anitha</td></tr> <tr><td>Ms. M.Madhuri</td></tr> <tr><td>Ms. Katherashala Swetha</td></tr> <tr><td>Ms. Velpula sumalatha</td></tr> <tr><td>Mr. Bingi Raju</td></tr> </table>		Mr. S Naresh Kumar	Ms. B. Swathi	Dr. Sasanko Shekhar Gantayat	Mr. Md Sallauddin	Dr. Mathivanan	Mr. Y Srikanth	Ms. N Shilpa	Dr. Rishabh Mittal (Coordinator)	Dr. R. Prashant Kumar	Mr. Ankushavali MD	Mr. B Viswanath	Ms. Sujitha Reddy	Ms. A. Anitha	Ms. M.Madhuri	Ms. Katherashala Swetha	Ms. Velpula sumalatha	Mr. Bingi Raju
Mr. S Naresh Kumar																				
Ms. B. Swathi																				
Dr. Sasanko Shekhar Gantayat																				
Mr. Md Sallauddin																				
Dr. Mathivanan																				
Mr. Y Srikanth																				
Ms. N Shilpa																				
Dr. Rishabh Mittal (Coordinator)																				
Dr. R. Prashant Kumar																				
Mr. Ankushavali MD																				
Mr. B Viswanath																				
Ms. Sujitha Reddy																				
Ms. A. Anitha																				
Ms. M.Madhuri																				
Ms. Katherashala Swetha																				
Ms. Velpula sumalatha																				
Mr. Bingi Raju																				
Course Code	23CS002PC304	Course Title	AI Assisted Coding																	
Year/Sem	III/II	Regulation	R23																	
Date and Day of Assignment	Week3 – Wednesday	Time(s)	23CSBTB01 To 23CSBTB52																	
Duration	2 Hours	Applicable to Batches	All batches																	
AssignmentNumber: 6.3(Present assignment number)/24(Total number of assignments)																				
Q.No.	Question	Expected Time to complete																		
1	Lab 6: AI-Based Code Completion – Classes, Loops, and Conditionals Lab Objectives <ul style="list-style-type: none"> • To explore AI-powered auto-completion features for core Python constructs such as classes, loops, and conditional statements. • To analyze how AI tools suggest logic for object-oriented programming and control structures. 	Week3 - Wednesday																		

- To evaluate the correctness, readability, and completeness of AI-generated Python code.

Lab Outcomes (LOs)

After completing this lab, students will be able to:

- Use AI tools to generate and complete Python class definitions and methods.
- Understand and assess AI-suggested loop constructs for iterative tasks.
- Generate and evaluate conditional statements using AI-driven prompts.
- Critically analyze AI-assisted code for correctness, clarity, and efficiency.

Task Description #1: Classes (Student Class)

Scenario

You are developing a simple student information management module.

Task

- Use an AI tool (GitHub Copilot / Cursor AI / Gemini) to complete a Student class.
- The class should include attributes such as name, roll number, and branch.
- Add a method display_details() to print student information.
- Execute the code and verify the output.
- Analyze the code generated by the AI tool for correctness and clarity.

Expected Output #1

•

```

1 class Student:
2     def __init__(self, name, roll_number, branch):
3         self.name = name
4         self.roll_number = roll_number
5         self.branch = branch
6
7     def display_details(self):
8         print(f"Name: {self.name}")
9         print(f"Roll Number: {self.roll_number}")
10        print(f"Branch: {self.branch}")
11
12 # Sample object creation
13 student1 = Student("Karthi", "2303A51801", "Computer Science")
14 student1.display_details()

```

```

PS C:\MY PYTHON> & C:/Users/KARTEEK/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/MY PYTHON/ai.py"
Name: Karthi
Roll Number: 2303A51801
Branch: Computer Science

```

Task Description #2: Loops (Multiples of a Number)

Scenario

You are writing a utility function to display multiples of a given number.

Task

- Prompt the AI tool to generate a function that prints the first 10 multiples of a given number using a loop.
- Analyze the generated loop logic.
- Ask the AI to generate the same functionality using another controlled looping structure (e.g., while instead of for).

Expected Output #2

The image shows a Visual Studio Code editor window with a file named `lab5.py` open. The code defines a `Student` class and a `print_multiples` function. The terminal at the bottom shows the command to run the script.

```

6         self.branch = branch
7
8     def display_details(self):
9         print(f"Name: {self.name}")
10        print(f"Roll Number: {self.roll_number}")
11        print(f"Branch: {self.branch}")
12
13    # Example usage
14    student1 = Student("Arun", "1735", "Computer Science")
15    student1.display_details()"""
16    # Write a Python function print_multiples(n) that prints first 10 multiples of n using for loop
17    def print_multiples(n):
18        for i in range(1, 11):
19            print(n * i)
20
21    # Example usage
22    print_multiples(5)

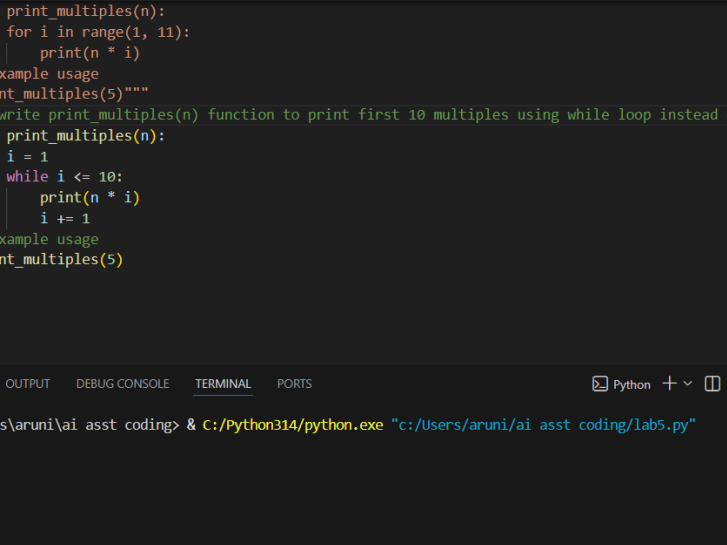
```

The terminal output shows the command to run the script:

```

PS C:\Users\aruni\ai asst coding> & C:/Python314/python.exe "c:/Users/aruni/ai asst coding/lab5.py"

```



The screenshot shows a VS Code editor with a Python file named `lab5.py`. The code defines a function `print_multiples(n)` that prints the first `n` multiples of `n` using a `while` loop. The function is called with `print_multiples(5)`. The terminal at the bottom shows the command to run the script, which has been executed successfully, resulting in the output of the function.

```

16 def print_multiples(n):
17     for i in range(1, 11):
18         print(n * i)
19 # Example usage
20 print_multiples(5)"""
21 #rewrite print_multiples(n) function to print first 10 multiples using while loop instead of f
22 def print_multiples(n):
23     i = 1
24     while i <= 10:
25         print(n * i)
26         i += 1
27 # Example usage
28 print_multiples(5)
29
30
31
32
33

```

```

PS C:\Users\aruni\ai asst coding> & C:/Python314/python.exe "c:/Users/aruni/ai asst coding/lab5.py"
5
10
15
20
25
30
35
40
45
50

```

Task Description #3: Conditional Statements (Age Classification)

Scenario

You are building a basic classification system based on age.

Task

- Ask the AI tool to generate nested if-elif-else conditional statements to classify age groups (e.g., child, teenager, adult, senior).
- Analyze the generated conditions and logic.
- Ask the AI to generate the same classification using alternative conditional structures (e.g.,

simplified conditions or dictionary-based logic).

Expected Output #3

-

```
29 #Create Python function classify_age(age) with nested if-elif-else to classify: child(<13), te
30 def classify_age(age):
31     if age < 13:
32         return "Child"
33     else:
34         if 13 <= age <= 19:
35             return "Teenager"
36         else:
37             if 20 <= age <= 59:
38                 return "Adult"
39             else:
40                 return "Senior"
41 # Example usage
42 print(classify_age(10)) # Output: Child
43 print(classify_age(16)) # Output: Teenager
44 print(classify_age(25)) # Output: Adult
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
```

```
45 #Rewrite classify_age(age) using dictionary mapping instead of if-elif statements
46 def classify_age(age):
47     age_groups = {
48         "Child": range(0, 13),
49         "Teenager": range(13, 20),
50         "Adult": range(20, 60),
51         "Senior": range(60, 150) # Assuming max age as 150
52     }
53
54     for group, age_range in age_groups.items():
55         if age in age_range:
56             return group
57 # Example usage
58 print(classify_age(10)) # Output: Child
59 print(classify_age(16)) # Output: Teenager
60 print(classify_age(25)) # Output: Adult
61 print(classify_age(70)) # Output: Senior
```

```
PS C:\Users\aruni\ai asst coding> C:/Python314/python.exe "c:/Users/aruni/ai asst coding/lab5.py"
Child
Teenager
Adult
```

```
PS C:\Users\aruni\ai asst coding> C:/Python314/python.exe "c:/Users/aruni/ai asst coding/lab5.py"
Child
Teenager
Adult
Senior
```

Task Description #4: For and While Loops (Sum of First n Numbers)

Scenario

You need to calculate the sum of the first n natural numbers.

Task

- Use AI assistance to generate a sum_to_n() function using a for loop.
- Analyze the generated code.
- Ask the AI to suggest an alternative implementation using a while loop or a mathematical formula.

Expected Output #4

-

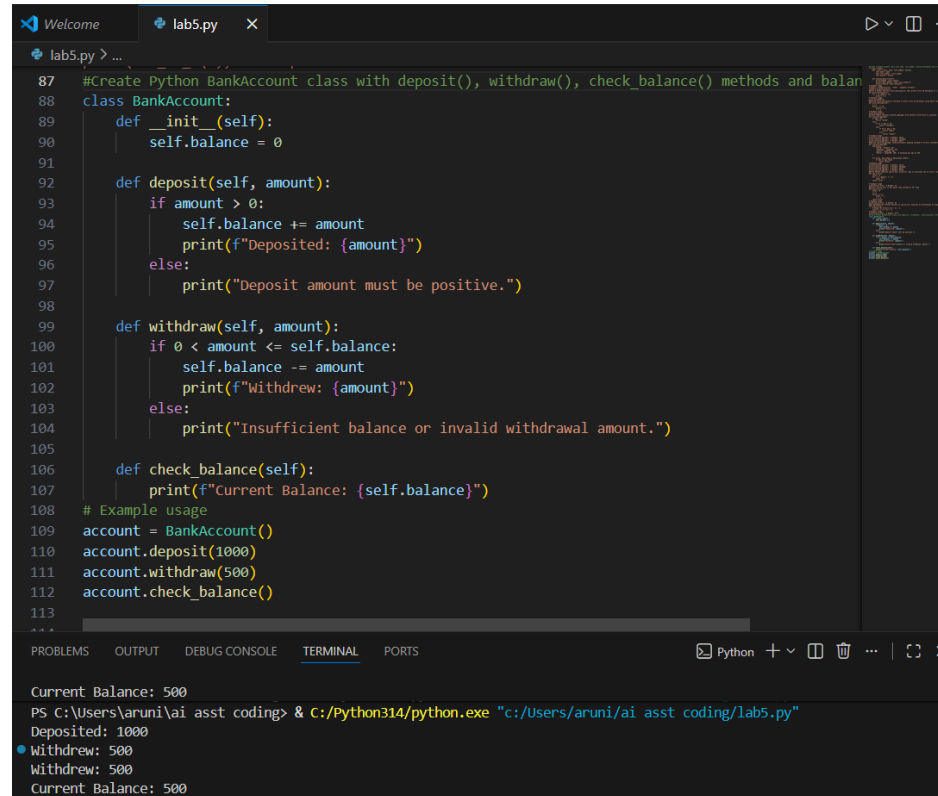
You are designing a basic banking application.

Task

- Use AI tools to generate a Bank Account class with methods such as deposit(), withdraw(), and check_balance().
- Analyze the AI-generated class structure and logic.
- Add meaningful comments and explain the working of the code.

Expected Output #5

•



```
87 #Create Python BankAccount class with deposit(), withdraw(), check_balance() methods and balan
88 class BankAccount:
89     def __init__(self):
90         self.balance = 0
91
92     def deposit(self, amount):
93         if amount > 0:
94             self.balance += amount
95             print(f"Deposited: {amount}")
96         else:
97             print("Deposit amount must be positive.")
98
99     def withdraw(self, amount):
100         if 0 < amount <= self.balance:
101             self.balance -= amount
102             print(f"Withdrew: {amount}")
103         else:
104             print("Insufficient balance or invalid withdrawal amount.")
105
106     def check_balance(self):
107         print(f"Current Balance: {self.balance}")
108
109 # Example usage
110 account = BankAccount()
111 account.deposit(1000)
112 account.withdraw(500)
113 account.check_balance()
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

```
Current Balance: 500
PS C:\Users\aruni\ai asst coding> C:/Python314/python.exe "c:/Users/aruni/ai asst coding/lab5.py"
Deposited: 1000
Withdrawn: 500
Withdrawn: 500
Current Balance: 500
```

Note: Report should be submitted as a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots.