WEEK 7 CRITICAL THINKING

https://github.com/pablicof90/AI-/upload/main

Sunday 12/29/24

This program creates a dictionary of 5 classes. The user will be able to search for the course name, number, instructor, and time. The output will contain key paired values.

‘’’

Write a program that creates a dictionary containing course name and numbers The dictionary should have the following key-value pairs:

Pseudocode

\_Initialize\_

input

class name

class number

Add class

Remove class

List class

output

print("Try programiz.pro")

class Study:

def \_\_init\_\_(frank):

frank.courses = {} # Dictionary to store courses, with course number as key

def add\_course(frank):

course\_number = input("Enter course number: ")

if course\_number in frank.courses:

print("Course with this number already exists.")

return

course\_name = input("Enter course name: ")

frank.courses[course\_number] = course\_name

print(f"Course '{course\_name}' with number '{course\_number}' added.")

def remove\_course(frank):

course\_number = input("Enter course number to remove: ")

if course\_number in frank.courses:

del frank.courses[course\_number]

print(f"Course with number '{course\_number}' removed.")

else:

print("Course not found.")

def display\_menu(frank):

print("\nCourse Management Menu")

print("1. Add Course")

print("2. Remove Course")

print("3. List Courses")

print("4. Exit")

def list\_courses(frank):

if not frank.courses:

print("No courses added yet.")

return

print("\nCourses:")

for number, name in frank.courses.items():

print(f"{number}: {name}")

def course\_input(frank):

while True:

frank.display\_menu()

choice = input("Enter your choice (1-4): ")

if choice == '1':

frank.add\_course()

elif choice == '2':

framk.remove\_course()

elif choice == '3':

frank.list\_courses()

elif choice == '4':

print("Exiting...")

break

else:

print("Invalid choice. Please try again.")

# Example usage

if \_\_name\_\_ == "\_\_main\_\_":

study = Study()

study.course\_input()

print("4. Exit")

A screenshot of a computer program

Description automatically generated

A blue square with white text

Description automatically generated with medium confidence

| **Key-Value Pairs: Course Name and Course Number** | |
| --- | --- |
| **Course Name(key)** | **Course Number (value)** |
| Principles of Programming | CSC101 |
| Math for Computer Science | CSC102 |
| Ethics | CSC103 |
| Computer Networking | NET110 |
| Leadership in Computer Science | COM241 |

The program should also create a dictionary containing course numbers and the names of the instructors that teach each course. The dictionary should have the following key-value pairs:

Pseudocode

\_Initialize\_

input

class number

class instructor

Add class

Remove class

List class

Output

Pseudocode

\_Initialize\_

input

class name

class number

Add class

Remove class

List class

output

print("4. Exit")

Pseudocode

\_Initialize\_

input

class name

class number

Add class

Remove class

List class

output

print("4. Exit")

A screenshot of a computer program

Description automatically generated

A blue square with white text

Description automatically generated with medium confidence

| **Key-Value Pairs: Instructors** | |
| --- | --- |
| **Course Number (key)** | **Instructor (value)** |
| CSC101 | Haynes |
| CSC102 | Alvarado |
| CSC103 | Rich |
| NET110 | Burke |
| COM241 | Lee |

The program should also create a dictionary containing course numbers and the meeting times of each course. The dictionary should have the following key-value pairs:

Pseudocode

\_Initialize\_

input

class number

class time

Add class

Remove class

List class

Output

# Online Python compiler (interpreter) to run Python online.

# Write Python 3 code in this online editor and run it.

print("Try programiz.pro")

class Study:

def \_\_init\_\_(frank):

frank.courses = {} # Dictionary to store courses, with course number as key

def add\_course(frank):

course\_number = input("Enter course number: ")

if course\_number in frank.courses:

print("Course with this number already exists.")

return

course\_time = input("Enter course time: ")

frank.courses[course\_number] = course\_time

print(f"Course '{course\_time}' with number '{course\_number}' added.")

def remove\_course(frank):

course\_number = input("Enter course number to remove: ")

if course\_number in frank.courses:

del frank.courses[course\_number]

print(f"Course with number '{course\_number}' removed.")

else:

print("Course not found.")

def display\_menu(frank):

print("\nCourse Management Menu")

print("1. Add Course")

print("2. Remove Course")

print("3. List Courses")

print("4. Exit")

def list\_courses(frank):

if not frank.courses:

print("No courses added yet.")

return

print("\nCourses:")

for number, name in frank.courses.items():

print(f"{number}: {name}")

def course\_input(frank):

while True:

frank.display\_menu()

choice = input("Enter your choice (1-4): ")

if choice == '1':

frank.add\_course()

elif choice == '2':

framk.remove\_course()

elif choice == '3':

frank.list\_courses()

elif choice == '4':

print("Exiting...")

break

else:

print("Invalid choice. Please try again.")

# Example usage

if \_\_name\_\_ == "\_\_main\_\_":

study = Study()

study.course\_input()

A screenshot of a computer program

Description automatically generated

A blue square with white text

Description automatically generated with medium confidence

|  |  |
| --- | --- |
| **Key-Value Pairs: Meeting Time** | |
| **Course Number (key)** | **Meeting Time (value)** |
| CSC101 | 8:00 a.m. |
| CSC102 | 9:00 a.m. |
| CSC103 | 10:00 a.m. |
| NET110 | 11:00 a.m. |
| COM241 | 1:00 p.m. |