Coding Style Requirements

Coding Style

Rule	Bad	Good	Penalty
Add a space after commas	`int num1,num2;`	`int num1, num2;`	-2
Add a space between operators and operands	`sum=5+3;`	`sum = 5 + 3;`	-2

Align the { } vertically (new line style)	`if(num > 0){`
	
	`}`
		`if(num > 0)`
		`{`
		
		`}`
			Graded 0

Add a block after `if`, `while` and `for` when they contain one statement	`if(num > 0)`
	`  x++;`
		`if(num > 0)`
		`{`
		`  x++;`
		`}`
			Graded 0

Indent your code appropriately in blocks and switch statements	`if(num > 0)`
	`{`
	`x++;`
	`}`
		`if(num > 0)`
		`{`
		`  x++;`
		`}`
			Graded 0

Unnecessary `continue` statements will be penalized	`while(k > 5)`
	`{`
	`  if(k > 3)`
	`  {`
	`  i++;`
	`  }`
	`  else`
	`  {`
	`  continue;`
	`  }`
	`}`
		`while(k > 5)`
		`{`
		`  if(k > 3)`
		`  {`
		`  i++;`
		`  }`
		`}`
			-10

Variable Naming

Rule	Bad	Good	Penalty
- Single letter names not allowed except for loop counters or temporary variables	`int n;`	`int number;`	-3
- Should always start with a lowercase	`float Ratio;`	`float ratio;`	-3
- Use abbreviations when they can be understood			
- number	`n`	`num`	-3
- customer first name	`n`	`firstName`	-3
	`name`	`custFirstName`	-3
- quantity	`q, qt`	`qty, quantity`	-3
- sentence	`s, sent`	`sentence`	-3
- number of students	`n, nos`	`numStudents, number`	-3

C - Assignment 08 (100 pts)

Instructions

- **Watch** the Module 09 – Formatted IO videos before starting this assignment.
- **Pseudocode not required.**
- Submit only one file containing all of your code. Multiple files not allowed!
- The knowledge required for this assignment is in the textbook / slides and your notes. **You are not allowed to use any additional resources, websites, external help... Doing so will be considered plagiarism and / or cheating!** If you need help, talk to a CIS tutor or to the professor.
- But you are allowed to assist your friends during the lab session in the presence of the professor.
- If you are having problems unrelated to the assignment (for example, you are unable to find a specific function in your IDE, …), you can go ahead and check it on the Internet.
- Do not use functions/techniques that have not been covered in class yet. Doing so might be mistaken for plagiarism and/or defeats the purpose of the exercise and will be graded 0.
- You are required to perform at least **one submission at the end of the lab session**, even if it's incomplete. You can complete it later and resubmit it; the latest one will be graded.

Exercise 01 (5 pts)

Create a structure `course` that is made from the following fields:

- Department (string, 15 characters)
- Course number (integer)
- Course title (string, 30 characters)
- Credits (short)

Complete with the required `typdef`.

Exercise 02 (15 pts)

Write a function `inputCourse()` which receives a `course` pointer and allows the user to input its information taking the size of each field into account. Test the function, but don't include the testing code in your homework.

Exercise 03 (20 pts)

Write a function `printCourse()` which receives a `course` pointer and prints all its fields, 1 field per line, using the following conditions:

- The labels ("Department", "Course Number", …) are left aligned
- The values are right aligned
- Course numbers must have leading zeroes

Test the function, but don't include the testing code in your homework.

Upload a screenshot of a sample output.

Use printf() formatted output so that your output should look like the following:

```
Department:                                 CIS
Course number:                             0835
Course title:           Cyberspace and Society
Credits:                                       3
```

Exercise 04 (20 pts)

Write a function `printCourseRow()` which receives a `course` pointer and prints all its fields as a single row. Use proper formatting so that when we print 2 or more courses as rows, the same members align below each other. Test the function, but don't include the testing code in your homework.

Upload a screenshot of a sample output.

Use printf() formatted output so that your output should look like the following:

```
CIS             0835     Cyberspace and Society                    3
CIS             1057     Computer programming in C                 4
```

Exercise 05 (10 pts)

Write a function `inputAllCourses()` which receives an **array of `course` pointers** and the array's size, then allows the user to input all courses in the array by calling `inputCourse()`

Exercise 06 (10 pts)

Write a function `printAllCourses()` which receives an **array of `course` pointers** and the array's size, then prints all courses in the array by calling `printCourseRow()`

Exercise 07 (20 pts)

Write a `main()` function using the following requirements:

- Define a `SIZE` constant (the value is irrelevant, but for testing, you may want to keep it small enough – no bigger than 5)
- Create an array of course pointers using `SIZE`
- Dynamically allocate each element of the array
- Call `inputAllCourses()`
- Call `printAllCourses()`