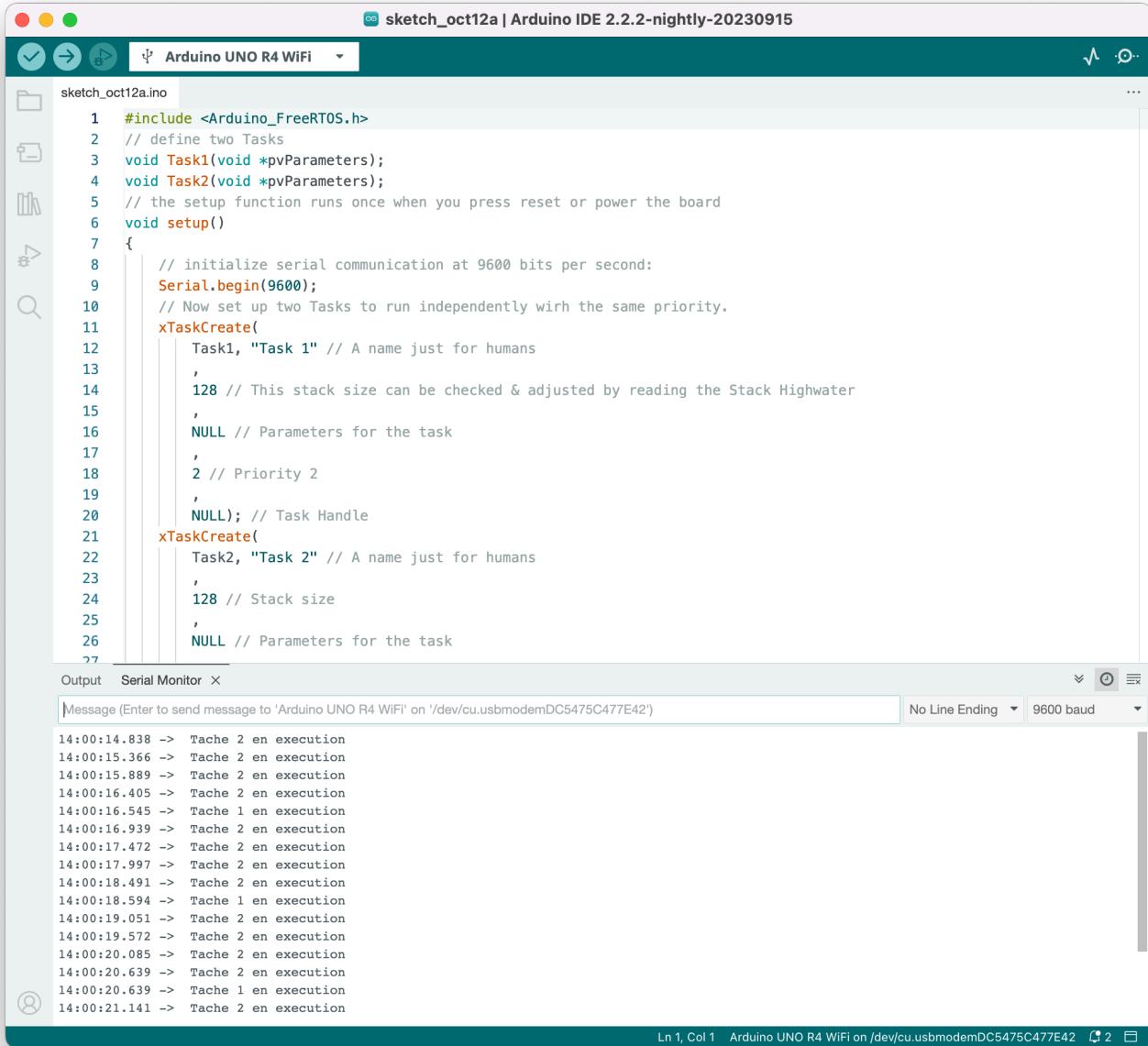


Programmation Multitâches – Rendu TP03

Exercice 3.1

Le code donné dans l'énoncé nous permet d'obtenir le résultat suivant.



```
#include <Arduino_FreeRTOS.h>
// define two Tasks
void Task1(void *pvParameters);
void Task2(void *pvParameters);
// the setup function runs once when you press reset or power the board
void setup()
{
    // initialize serial communication at 9600 bits per second:
    Serial.begin(9600);
    // Now set up two Tasks to run independently with the same priority.
    xTaskCreate(
        Task1, "Task 1" // A name just for humans
        ,
        128 // This stack size can be checked & adjusted by reading the Stack Highwater
        ,
        NULL // Parameters for the task
        ,
        2 // Priority 2
        ,
        NULL); // Task Handle
    xTaskCreate(
        Task2, "Task 2" // A name just for humans
        ,
        128 // Stack size
        ,
        NULL // Parameters for the task
    );
}
// Task1
void Task1(void *pvParameters)
{
    while(1)
    {
        // Task1 code here
        // ...
    }
}
// Task2
void Task2(void *pvParameters)
{
    while(1)
    {
        // Task2 code here
        // ...
    }
}
```

Output Serial Monitor

Message (Enter to send message to 'Arduino Uno R4 WiFi' on '/dev/cu.usbmodemDC5475C477E42')

No Line Ending 9600 baud

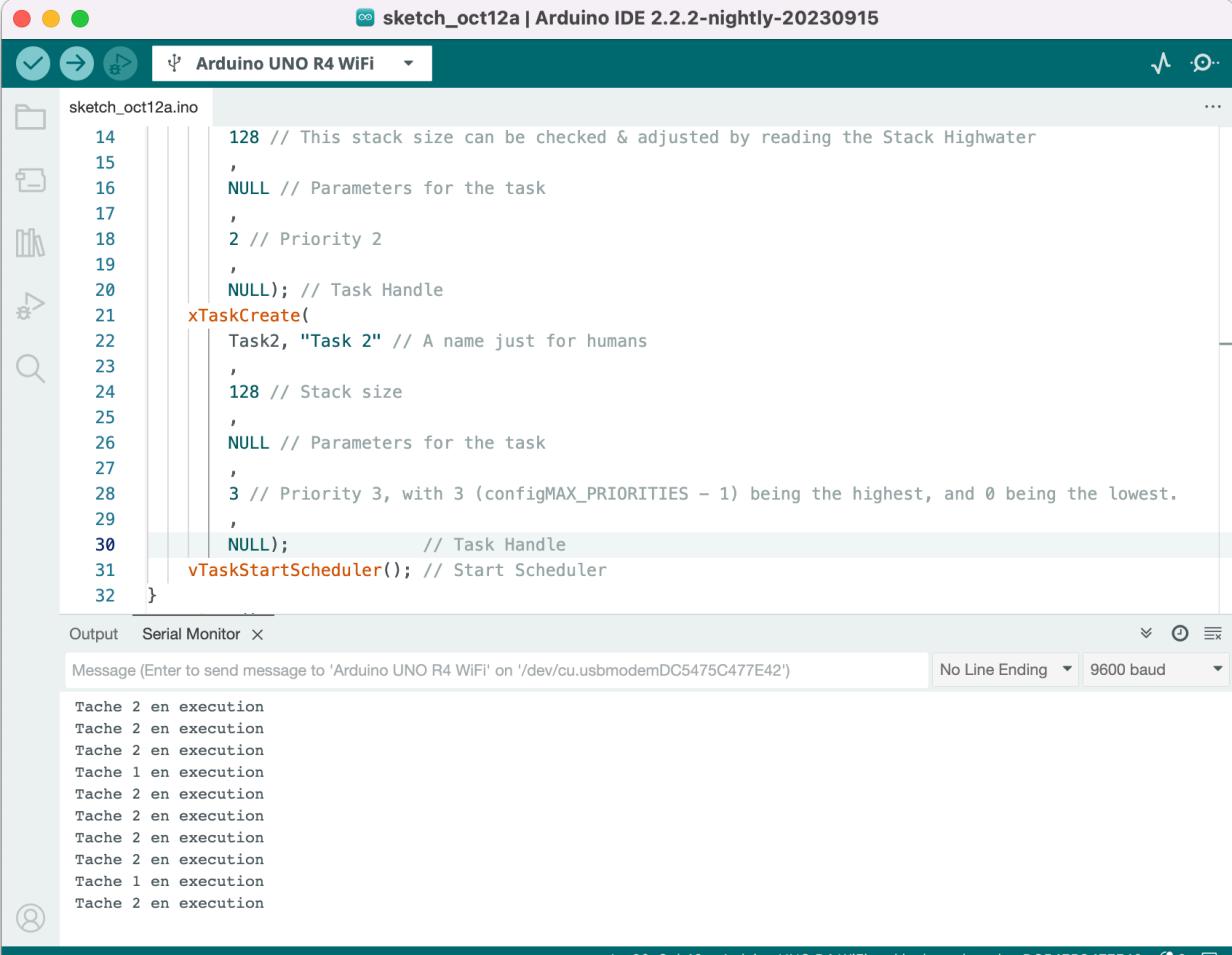
```
14:00:14.838 -> Tache 2 en execution
14:00:15.366 -> Tache 2 en execution
14:00:15.889 -> Tache 2 en execution
14:00:16.405 -> Tache 2 en execution
14:00:16.545 -> Tache 1 en execution
14:00:16.939 -> Tache 2 en execution
14:00:17.472 -> Tache 2 en execution
14:00:17.997 -> Tache 2 en execution
14:00:18.491 -> Tache 2 en execution
14:00:18.594 -> Tache 1 en execution
14:00:19.051 -> Tache 2 en execution
14:00:19.572 -> Tache 2 en execution
14:00:20.085 -> Tache 2 en execution
14:00:20.639 -> Tache 2 en execution
14:00:20.639 -> Tache 1 en execution
14:00:21.141 -> Tache 2 en execution
```

Ln 1, Col 1 Arduino Uno R4 WiFi on /dev/cu.usbmodemDC5475C477E42

Nous pouvons constater que la tâche 1 va afficher une fois « Tâche 1 en exécution » toutes les 2000ms puis se mettra en attente pendant cette durée. Il en est de même pour la tâche n°2, qui elle sera mise en attente (**Blocked**) toutes les 500ms avant d'être mise en état **Ready**, soit 4 fois plus souvent que la tâche n°1. C'est pourquoi nous pouvons voir « Tâche 2 en exécution » 4 fois plus souvent sur notre **Serial Monitor**.

Exercice 3.2

La première tâche réalisée est la tâche n°2. Ce comportement n'est pas étonnant dans la mesure où cette tâche possède la priorité la plus élevée.



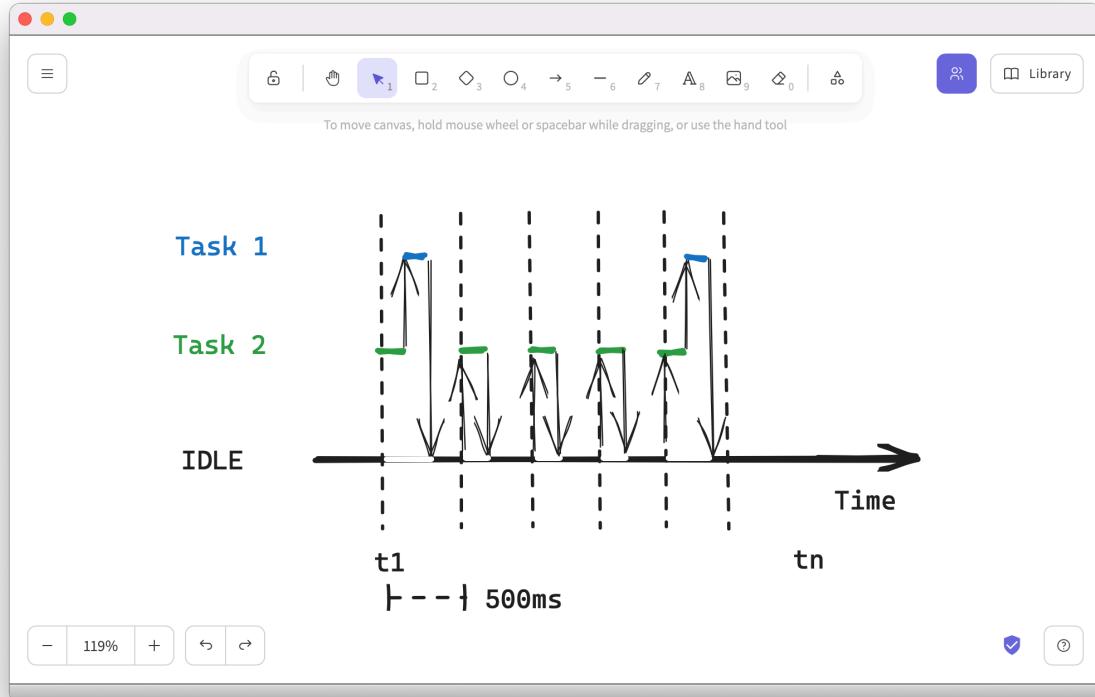
The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** sketch_oct12a | Arduino IDE 2.2.2-nightly-20230915
- Sketch:** sketch_oct12a.ino
- Code:**

```
14 128 // This stack size can be checked & adjusted by reading the Stack Highwater
15 ,
16 NULL // Parameters for the task
17 ,
18 2 // Priority 2
19 ,
20 NULL); // Task Handle
21 xTaskCreate(
22 Task2, "Task 2" // A name just for humans
23 ,
24 128 // Stack size
25 ,
26 NULL // Parameters for the task
27 ,
28 3 // Priority 3, with 3 (configMAX_PRIORITIES - 1) being the highest, and 0 being the lowest.
29 ,
30 NULL); // Task Handle
31 vTaskStartScheduler(); // Start Scheduler
32 }
```
- Output Tab:** Serial Monitor
- Serial Monitor Content:** Tache 2 en execution
Tache 2 en execution
Tache 2 en execution
Tache 1 en execution
Tache 2 en execution
Tache 2 en execution
Tache 2 en execution
Tache 2 en execution
Tache 1 en execution
Tache 2 en execution
- Bottom Status:** Ln 30, Col 42 Arduino Uno R4 WiFi on /dev/cu.usbmodemDC5475C477E42 4 2

Exercice 3.3

D'après le code donné en énoncé, la séquence d'exécution de ce dernier serait la suivante.



Exercice 3.4

En effet, en commentant ces lignes, nous obtenons un résultat similaire.

```

sketch_oct12a.ino
41 {
42     // prints Tache 1 to the serial monitor
43     volatile uint32_t cnt;
44     const TickType_t xDelay2000ms = pdMS_TO_TICKS(2000); // convertit en ticks une
45     for (;;) { // A Task shall never return
46         Serial.println(" Tache 1 en execution");
47         for(cnt=0;cnt<9000000;cnt++) {};
48         // delay(2000); delay en arduino encapsule vtaskdelay de FreeRTOS
49         // vTaskDelay(xDelay2000ms); // la tache 1 passe en mode bloqué pendant 2 s
50         // liberté à la tache 2 de priorité inférieure de s'exécuter
51     }
52 }
53 void Task2(void *pvParameters) // This is a Task 2.
54 {
55 }
```

Message (Enter to send message to 'Arduino Uno R4 WiFi' on '/dev/cu.usbmodemDC5475C477E42')
No Line Ending 9600 baud

Tache 2 en execution
Tache 2 en execution
Tache 2 en execution
Tache 1 en exec Tache 2 en execution
Tache 2 en execution
Tache 2 en execution
Tache 2 en execution
Tache 1 en execution
Tache 2 en execution
...

La tâche 2 ayant une priorité plus haute, et l'utilisation de la boucle `for` n'étant pas aussi précis (de loin) que l'utilisation de la fonction `vTaskDelay()`, à certains moments, la boucle `for` va s'exécuter de façon sensiblement plus rapide ou plus lente, perturbant donc l'affichage des chaînes de caractères.

Exercice 3.5

Le fichier édité est le suivant.

Le problème ne semble plus apparaître.

The screenshot shows the Arduino IDE interface with the following details:

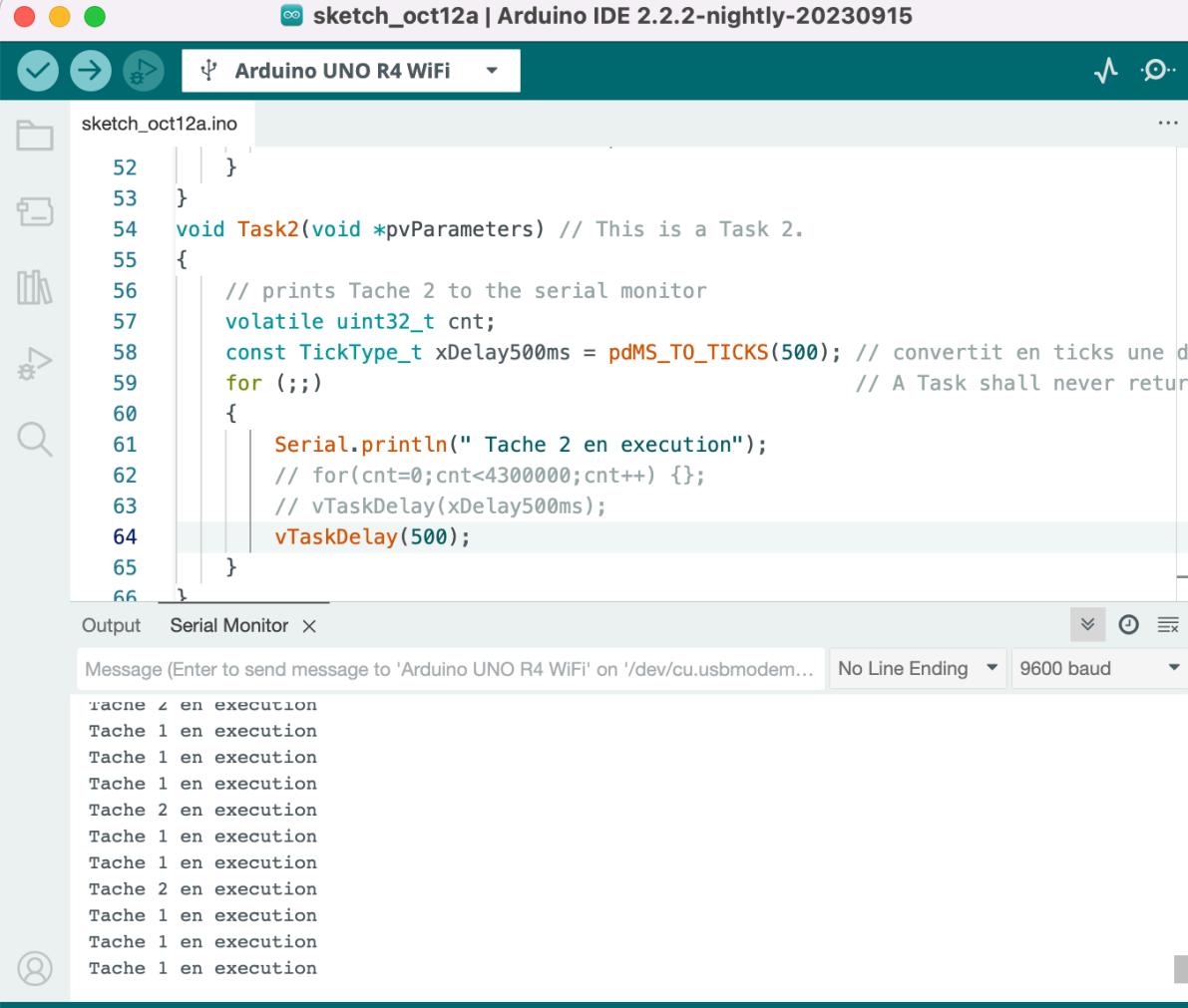
- Title Bar:** sketch_oct12a | Arduino IDE 2.2.2-nightly-20230915
- Toolbar:** Includes icons for file operations (New, Open, Save, Print, Find, Copy, Paste, Undo, Redo), a search bar, and a help icon.
- Sketch List:** sketch_oct12a.ino
- Code Editor:** Displays C++ code for tasks. Task 1 is defined with a loop that prints "Tache 1 en execution" to the serial monitor every 2000ms. Task 2 is defined as a function that prints "Tache 2 en execution".

```
41 }  
42 // prints Tache 1 to the serial monitor  
43 volatile uint32_t cnt;  
44 const TickType_t xDelay2000ms = pdMS_TO_TICKS(2000); // convertit en ticks une  
45 for (;;) // A Task shall never return  
46 {  
47     Serial.println(" Tache 1 en execution");  
48     for(cnt=0;cnt<9000000;cnt++) {};  
49     // delay(2000); delay in arduino encapsule vtaskdelay de Freertos  
50     // vTaskDelay(xDelay2000ms); // la tache 1 passe en mode bloqué pendant 2 s  
51     // liberté à la tache 2 de priorité inférieure de s'exécuter  
52 }  
53 }  
54 void Task2(void *pvParameters) // This is a Task 2.  
55 }
```
- Serial Monitor:** Set to "Serial Monitor" tab. The message input field contains "Message (Enter to send message to 'Arduino Uno R4 WiFi' on '/dev/cu.usbmodemDC5475C477E42' No Line Ending)". The baud rate is set to 9600 baud. The output window shows the following text:

```
Tache 2 en execution  
Tache 2 en execution  
Tache 1 en execution  
Tache 2 en execution  
Tache 2 en execution  
Tache 2 en execution  
Tache 2 en execution  
Tache 1 en execution  
Tache 2 en execution  
Tache 2 en execution  
Tache 2 en execution
```

Exercice 3.6

Après modification de la ligne, nous observons le résultat suivant.



```
sketch_oct12a | Arduino IDE 2.2.2-nightly-20230915
Arduino UNO R4 WiFi

sketch_oct12a.ino

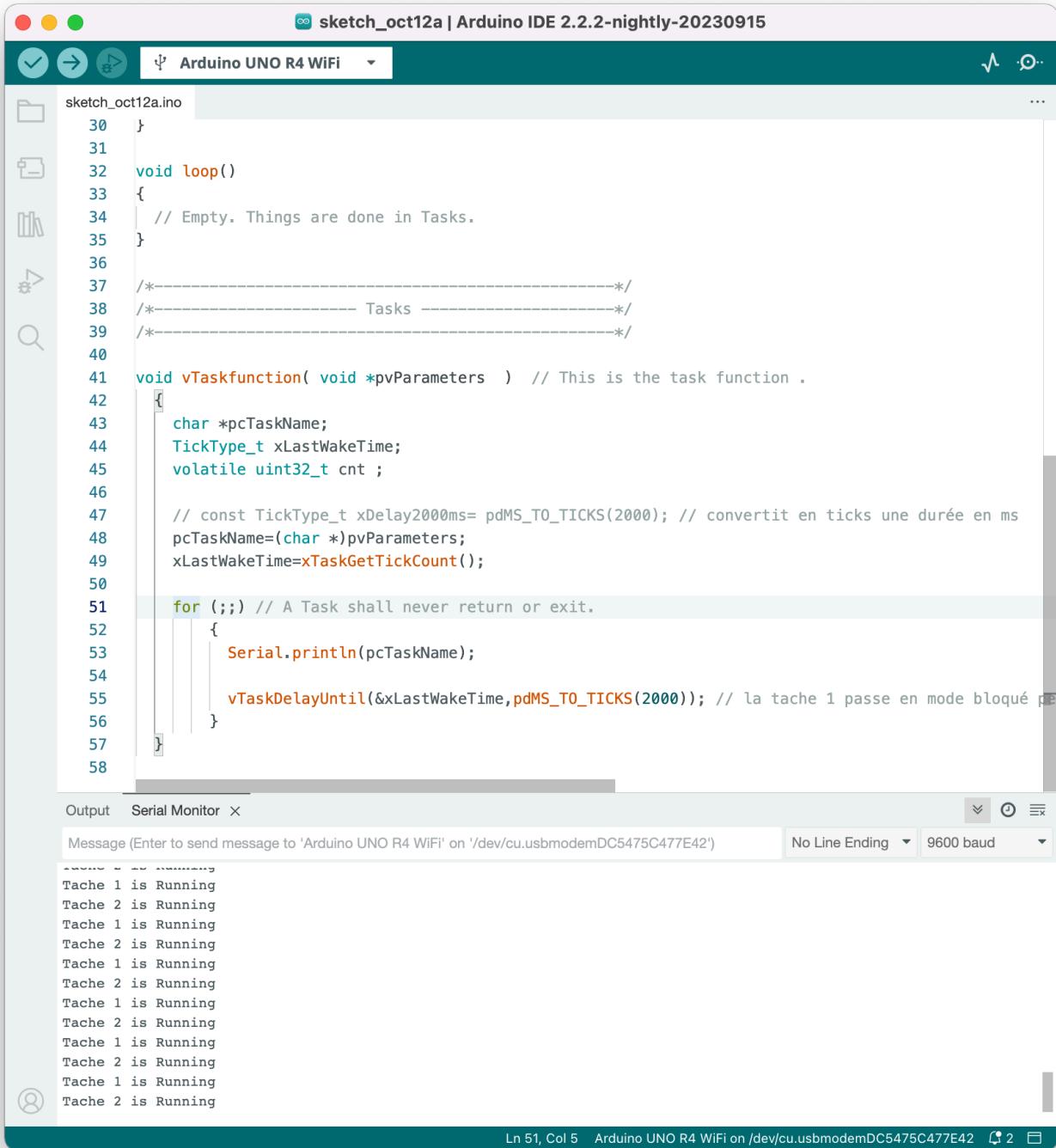
52     }
53 }
54 void Task2(void *pvParameters) // This is a Task 2.
55 {
56     // prints Tache 2 to the serial monitor
57     volatile uint32_t cnt;
58     const TickType_t xDelay500ms = pdMS_TO_TICKS(500); // convertit en ticks une d...
59     for (;;)                                // A Task shall never return
60     {
61         Serial.println(" Tache 2 en execution");
62         // for(cnt=0;cnt<4300000;cnt++);
63         // vTaskDelay(xDelay500ms);
64         vTaskDelay(500);
65     }
66 }

Output Serial Monitor ×
Message (Enter to send message to 'Arduino UNO R4 WiFi' on '/dev/cu.usbmodem...') No Line Ending 9600 baud
Tache 1 en execution
Tache 1 en execution
Tache 1 en execution
Tache 1 en execution
Tache 2 en execution
Tache 1 en execution
Tache 1 en execution
Tache 2 en execution
Tache 1 en execution
Tache 1 en execution
Tache 1 en execution
Ln 64, Col 25  Arduino UNO R4 WiFi on /dev/cu.usbmodemDC5475C477E42  ↻ 2
```

La séquence d'exécution est devenue plus lente, et on observe des irrégularités dans le comportement du programme. C'est normal, le `vTaskDelay(500)` attend la durée de **500 Tick**, et non **500ms**.

Exercice 3.7 & 3.8

Nous obtenons le résultat suivant.



The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** sketch_oct12a | Arduino IDE 2.2.2-nightly-20230915
- Toolbars:** Standard Arduino IDE toolbars for file operations (New, Open, Save, Print, etc.) and project management.
- Sketch List:** A sidebar showing the file sketch_oct12a.ino.
- Code Editor:** The main area contains C++ code for a task function. The code initializes variables, sets up a task with a delay of 2000ms, and enters a loop where it prints the task name and waits for the next tick.

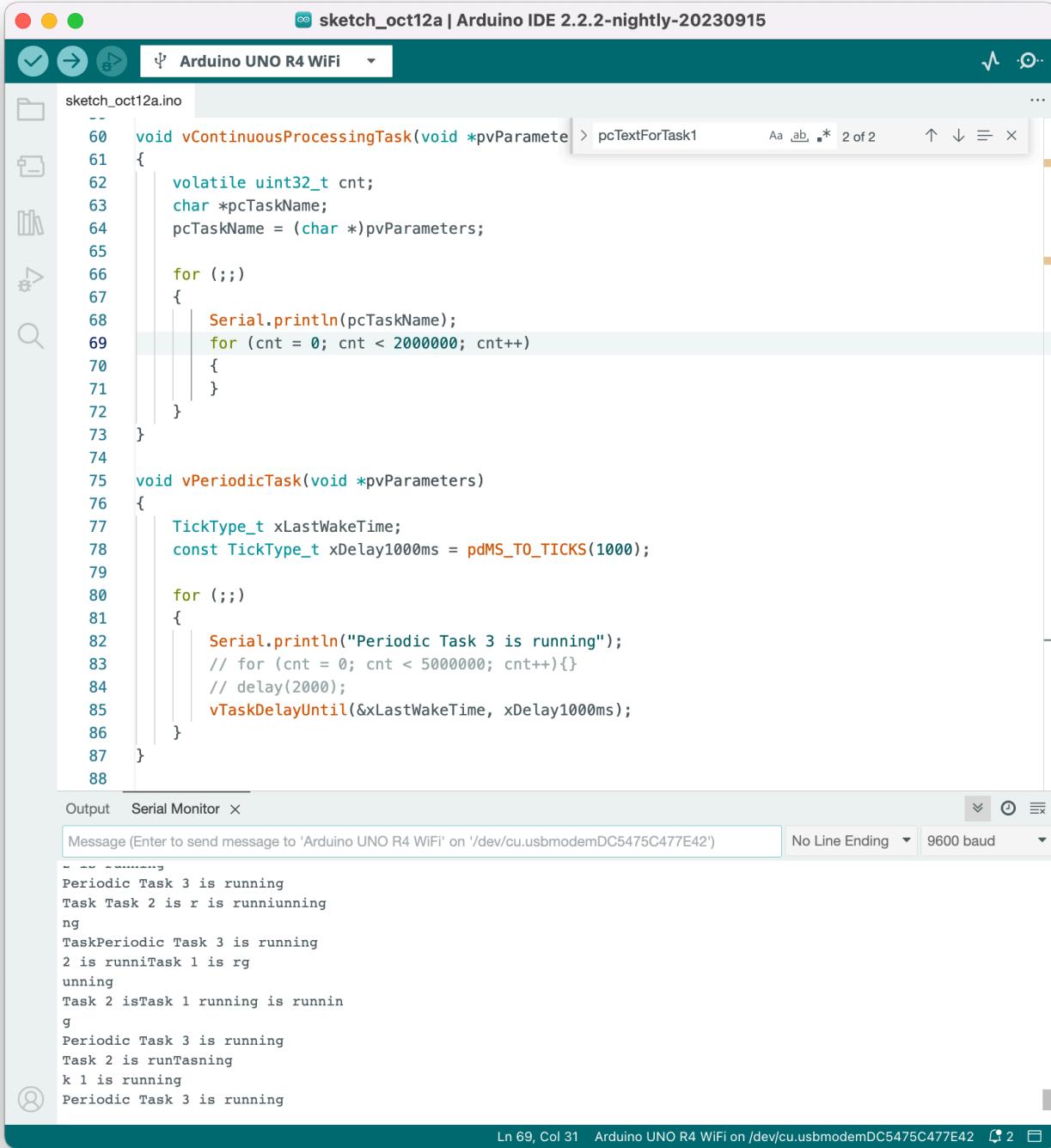
```
sketch_oct12a.ino
30 }
31
32 void loop()
33 {
34 // Empty. Things are done in Tasks.
35 }
36
37 /*-----*
38 *----- Tasks -----*
39 *-----*/
40
41 void vTaskfunction( void *pvParameters ) // This is the task function .
42 {
43     char *pcTaskName;
44     TickType_t xLastWakeTime;
45     volatile uint32_t cnt ;
46
47     // const TickType_t xDelay2000ms= pdMS_TO_TICKS(2000); // convertit en ticks une durée en ms
48     pcTaskName=(char *)pvParameters;
49     xLastWakeTime=xTaskGetTickCount();
50
51     for (;;) // A Task shall never return or exit.
52     {
53         Serial.println(pcTaskName);
54
55         vTaskDelayUntil(&xLastWakeTime,pdMS_TO_TICKS(2000)); // la tache 1 passe en mode bloqué p
56     }
57 }
58
```

- Output Tab:** Shows the serial port configuration: Message (Enter to send message to 'Arduino UNO R4 WiFi' on '/dev/cu.usbmodemDC5475C477E42'), No Line Ending, 9600 baud.
- Serial Monitor Tab:** Displays the output of the task function, showing alternating prints for "Tache 1 is Running" and "Tache 2 is Running".
- Status Bar:** Shows the current line (Ln 51), column (Col 5), and the connection information: Arduino UNO R4 WiFi on /dev/cu.usbmodemDC5475C477E42.

Nous pouvons observer une parfaite régularité d'exécution, contrairement à ce que nous pouvions constater dans l'exemple 6.

Exercice 3.9

Après avoir exécuté la tâche, nous obtenons le résultat suivant.



The screenshot shows the Arduino IDE interface with the sketch `sketch_oct12a.ino` open. The code defines three tasks: `vContinuousProcessingTask`, `vPeriodicTask`, and `vTaskDelayUntil`. The `vContinuousProcessingTask` prints the name of the task and a counter from 0 to 2,000,000. The `vPeriodicTask` prints "Periodic Task 3 is running" and delays until the next tick. The `vTaskDelayUntil` task is triggered by the periodic task, causing it to print "Task 2 is running" and "Task 1 is running". The serial monitor output shows the following sequence of messages:

```
Message (Enter to send message to 'Arduino UNO R4 WiFi' on '/dev/cu.usbmodemDC5475C477E42')
Periodic Task 3 is running
Task Task 2 is r is runniunning
ng
TaskPeriodic Task 3 is running
2 is runniTask 1 is rg
unning
Task 2 isTask 1 running is runnin
g
Periodic Task 3 is running
Task 2 is runTasning
k 1 is running
Periodic Task 3 is running
```

Il n'est pas étonnant. La tâche 3 est exécutée de façon périodique en suivant le nombre de ticks absolu. Elle s'exécute donc de façon régulière. Cependant, les deux autres tâches ne sont pas mises en état Blocked, et son déclenchées en décalé lorsque la tâche 3 devait s'exécuter en même temps qu'eux. (La priorité de la tâche 3 étant plus élevée, elle passe en première.)

Le modèle d'exécution présenté ensuite dans l'énoncé illustre parfaitement cette hypothèse.

Exercice 3.10

Il semble que la constante configUSE_IDLE_HOOK soit déjà mise à 1 dans le fichier FreeRTOSConfig.h.

```
" Press ? for help
.. (up a dir)
</Arduino_FreeRTOS/src/
► lib/
► portable/
Arduino_FreeRTOS.h
FreeRTOSConfig.h

20 #ifndef configUSE_PREEMPTION
21 #define configUSE_PREEMPTION (1)
22 #endif
23 #ifndef configUSE_PORT_OPTIMISED_TASK_SELECTION
24 #define configUSE_PORT_OPTIMISED_TASK_SELECTION (0)
25 #endif
26 #ifndef configUSE_TICKLESS_IDLE
27 #define configUSE_TICKLESS_IDLE (0)
28 #endif
29 #ifndef configUSE_IDLE_HOOK
30 #define configUSE_IDLE_HOOK (1)
31 #endif
32 #ifndef configUSE_MALLOC_FAILED_HOOK
33 #define configUSE_MALLOC_FAILED_HOOK (0)
34 #endif
35 #ifndef configUSE_DAEMON_TASK_STARTUP_HOOK
36 #define configUSE_DAEMON_TASK_STARTUP_HOOK (0)
37 #endif
38 #ifndef configUSE_TICK_HOOK
39 #define configUSE_TICK_HOOK (0)

N... ➤ FreeRTOSConfig.h 29/233
```

The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** sketch_oct12a.ino | Arduino Uno R4 WiFi | Arduino IDE 2.2.2-nightly-20230915
- File Menu:** File, Open, Save, Preferences, Help
- Tools Menu:** Tools, Board, Tools, Port, Tools, Sketch, Tools, Help
- Sketch Oct12a.ino Content:**

```
39  /*----- Tasks -----> pcTextForTask1
40  *----- Tasks -----*/
41  /*
42 void vApplicationIdleHook(void)
43 {
44     volatile uint32_t cnt;
45     ulIdleCycleCount++;
46     for(cnt=0;cnt<4710;cnt++);
47 }
48
49
50 void vTaskfunction( void *pvParameters ) // This is the task function .
51 {
52     char *pcTaskName;
53     TickType_t xLastWakeTime;
54     volatile uint32_t cnt ;
55
56
57     pcTaskName=(char *)pvParameters;
58     xLastWakeTime=xTaskGetTickCount();
59
60     for (;;) // A Task shall never return or exit.
61     {
62         Serial.println(pcTaskName);
63         Serial.println(ulIdleCycleCount);
64
65         vTaskDelayUntil(&xLastWakeTime,pdMS_TO_TICKS(2000));
66     }                                // la tache 2 passe alors en mode bloqué et laisse le temps res
67 }
```
- Output Serial Monitor:**

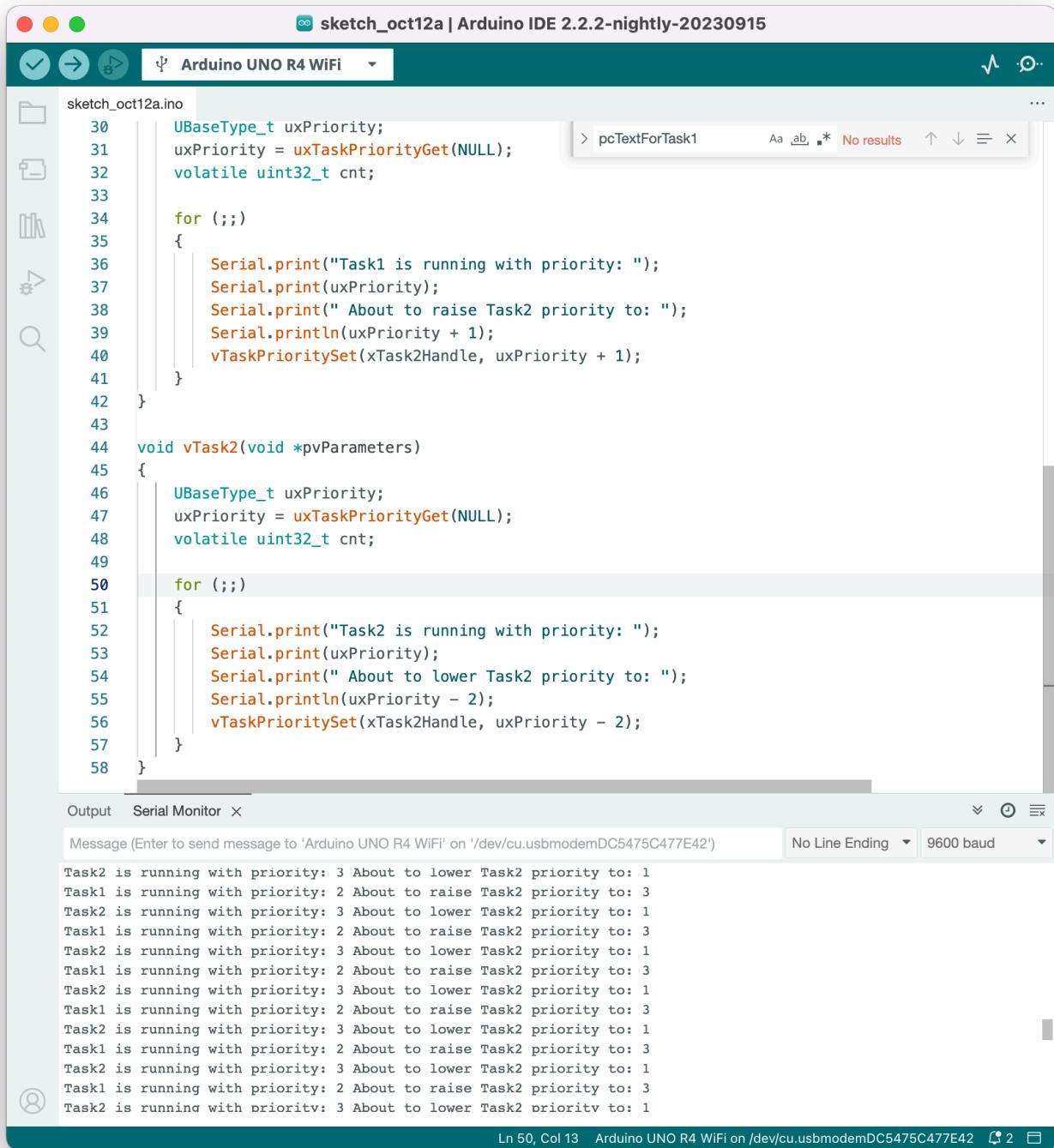
```
Tache 1 is Running
9888
Tache 2 is Running
9888
Tache 1 is Running
11864
Tache 2 is Running
11864
Tache 1 is Running
13839
Tache 2 is Running
13839
```
- Status Bar:** Ln 63, Col 44 | Arduino Uno R4 WiFi on /dev/cu.usbmodemDC5475C477E42 | 9600 baud

Nous pouvons constater que le compteur incrémenté dans `vApplicationIdleHook()` indiquant le nombre de ms écoulées en `Idle` est en constant augmentation entre l'exécution de la tâche 2 et la tâche 1, mais pas entre la tâche 1 et la tâche 2, indiquant donc que les deux tâches sont exécutées à la suite.

Exercice 3.11

Partie A

Nous obtenons le résultat suivant :



The screenshot shows the Arduino IDE interface with the sketch `sketch_oct12a.ino` open. The code implements two tasks, Task1 and Task2, using FreeRTOS API functions. Task1 runs at a higher priority (3) and prints its priority and a message to raise Task2's priority. Task2 runs at a lower priority (1) and prints its priority and a message to lower Task2's priority. The Serial Monitor output shows the tasks switching priorities as expected.

```
sketch_oct12a | Arduino IDE 2.2.2-nightly-20230915
Arduino UNO R4 WiFi

sketch_oct12a.ino

30 UBaseType_t uxPriority;
31 uxPriority = uxTaskPriorityGet(NULL);
32 volatile uint32_t cnt;
33
34 for (;;)
35 {
36     Serial.print("Task1 is running with priority: ");
37     Serial.print(uxPriority);
38     Serial.print(" About to raise Task2 priority to: ");
39     Serial.println(uxPriority + 1);
40     vTaskPrioritySet(xTask2Handle, uxPriority + 1);
41 }
42
43
44 void vTask2(void *pvParameters)
45 {
46     UBaseType_t uxPriority;
47     uxPriority = uxTaskPriorityGet(NULL);
48     volatile uint32_t cnt;
49
50     for (;;)
51     {
52         Serial.print("Task2 is running with priority: ");
53         Serial.print(uxPriority);
54         Serial.print(" About to lower Task2 priority to: ");
55         Serial.println(uxPriority - 2);
56         vTaskPrioritySet(xTask2Handle, uxPriority - 2);
57     }
58 }

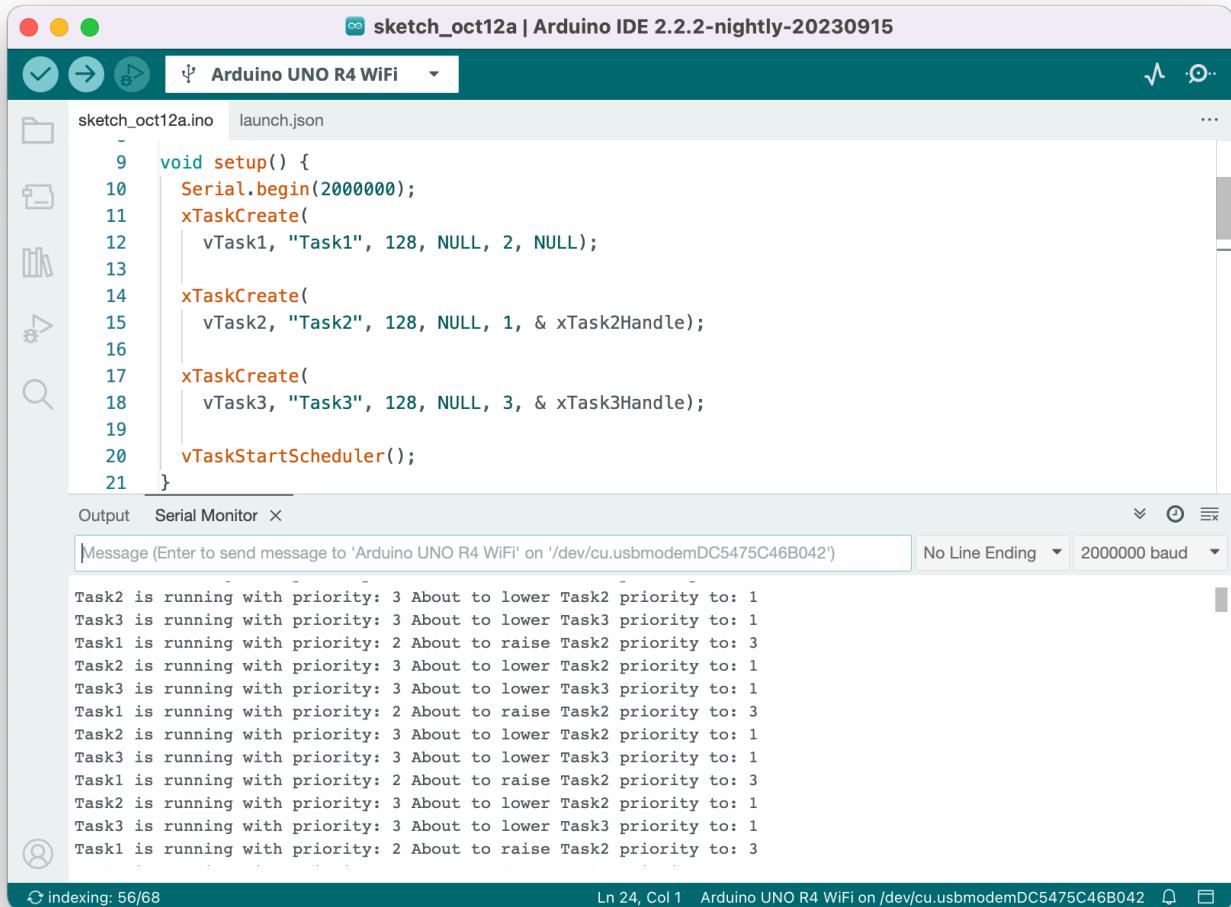
Output Serial Monitor ×
Message (Enter to send message to 'Arduino UNO R4 WiFi' on '/dev/cu.usbmodemDC5475C477E42')
No Line Ending 9600 baud
Task2 is running with priority: 3 About to lower Task2 priority to: 1
Task1 is running with priority: 2 About to raise Task2 priority to: 3
Task2 is running with priority: 3 About to lower Task2 priority to: 1
Task1 is running with priority: 2 About to raise Task2 priority to: 3
Task2 is running with priority: 3 About to lower Task2 priority to: 1
Task1 is running with priority: 2 About to raise Task2 priority to: 3
Task2 is running with priority: 3 About to lower Task2 priority to: 1
Task1 is running with priority: 2 About to raise Task2 priority to: 3
Task2 is running with priority: 3 About to lower Task2 priority to: 1
Task1 is running with priority: 2 About to raise Task2 priority to: 3
Task2 is running with priority: 3 About to lower Task2 priority to: 1
Task1 is running with priority: 2 About to raise Task2 priority to: 3
Task2 is running with priority: 3 About to lower Task2 priority to: 1
Ln 50, Col 13 Arduino UNO R4 WiFi on /dev/cu.usbmodemDC5475C477E42 4:2
```

Nous pouvons constater que le déroulement du programme s'effectue de façon régulière. La tâche 2 passe de la priorité 3 à la priorité 1, puis de 1 à 3 et ainsi de suite. La priorité de la tâche 1 n'étant pas changé, cette dernière s'exécute avant la tâche 2 quand cette dernière possède une priorité à 1.

Partie B

Pour obtenir l'ordre des tâches Task1, Task2, Task3, il faut définir une nouvelle tâche et jouer un peu sur les priorités.

On obtient le résultat suivant :



The screenshot shows the Arduino IDE interface with the sketch `sketch_oct12a.ino` open. The code defines three tasks using the FreeRTOS API:

```
void setup() {
    Serial.begin(2000000);
    xTaskCreate(
        vTask1, "Task1", 128, NULL, 2, NULL);
    xTaskCreate(
        vTask2, "Task2", 128, NULL, 1, &xTask2Handle);
    xTaskCreate(
        vTask3, "Task3", 128, NULL, 3, &xTask3Handle);
    vTaskStartScheduler();
}
```

The `Serial Monitor` window displays the following output, indicating the periodic switching of task priorities:

```
Task2 is running with priority: 3 About to lower Task2 priority to: 1
Task3 is running with priority: 3 About to lower Task3 priority to: 1
Task1 is running with priority: 2 About to raise Task2 priority to: 3
Task2 is running with priority: 3 About to lower Task2 priority to: 1
Task3 is running with priority: 3 About to lower Task3 priority to: 1
Task1 is running with priority: 2 About to raise Task2 priority to: 3
Task2 is running with priority: 3 About to lower Task2 priority to: 1
Task3 is running with priority: 3 About to lower Task3 priority to: 1
Task1 is running with priority: 2 About to raise Task2 priority to: 3
Task2 is running with priority: 3 About to lower Task2 priority to: 1
Task3 is running with priority: 3 About to lower Task3 priority to: 1
Task1 is running with priority: 2 About to raise Task2 priority to: 3
```

Le code des différentes tâches est le suivant :



The screenshot shows the Arduino IDE interface with the title bar "sketch_oct12a | Arduino IDE 2.2.2-nightly-20230915". The central area displays the following C code for three tasks:

```
void vTask1(void * pvParameters) {
    UBaseType_t uxPriority;
    uxPriority = uxTaskPriorityGet(NULL);
    volatile uint32_t cnt;

    for (;;) {
        Serial.print("Task1 is running with priority: ");
        Serial.print(uxPriority);
        Serial.print(" About to raise Task2 priority to: ");
        Serial.println(uxPriority + 1);
        vTaskPrioritySet(xTask2Handle, uxPriority + 1);
    }
}

void vTask2(void * pvParameters) {
    UBaseType_t uxPriority;
    uxPriority = uxTaskPriorityGet(NULL);
    volatile uint32_t cnt;

    for (;;) {
        Serial.print("Task2 is running with priority: ");
        Serial.print(uxPriority);
        Serial.print(" About to lower Task2 priority to: ");
        Serial.println(uxPriority - 2);
        vTaskPrioritySet(xTask3Handle, uxPriority + 1);
        vTaskPrioritySet(NULL, uxPriority - 2);
    }
}

void vTask3(void * pvParameters) {
    UBaseType_t uxPriority;
    uxPriority = uxTaskPriorityGet(NULL);
    volatile uint32_t cnt;

    for (;;) {
        Serial.print("Task3 is running with priority: ");
        Serial.print(uxPriority);
        Serial.print(" About to lower Task3 priority to: ");
        Serial.println(uxPriority - 2);
        vTaskPrioritySet(NULL, uxPriority - 2);
    }
}
```

The Arduino IDE status bar at the bottom indicates "Not connected. Select a board and a port to connect automatically." and "Ln 54, Col 1 Arduino Uno R4 WiFi on /dev/cu.usbmodemDC5475C46B042 [not connected]".

Exercice 3.12