Engenharia de Sistemas Informáticos
Sistemas Embebidos e de Tempo Real
Ano Letivo 2018/2019

Época Normal									
Semestral	☒ 1º Teste	**Anual**	**1º Chamada**	☐ 1º Teste	☐ 2º Teste	☐ Global			
	☐ 2º Teste		**2º Chamada**	☐					
Época Recurso	☐	**Época Especial**	☐		**Exame Especial** ☐				

Duração: 1h30min	**Tolerância:** 30 minutos	☐	**Com Consulta**
		☒	**Sem Consulta**
Docente: __Sofia Miranda da Silva Portela_____		**Data:** 06/06/2019	
Nome:_____		**Número:**_____	

Versão A
Grupo 1 (5 valores)
(cada resposta correta vale 0,5 valores e cada resposta errada desconta 0,2 valores no grupo)

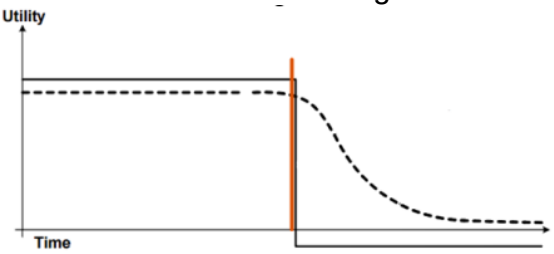
1. Verdadeiro ou Falso

	Os sistemas de informação gerem estruturas de dados complexas e são dependentes do tipo de hardware.
	Um escalonador preemptivo é aquele que permite substituir uma tarefa de menor prioridade por uma de prioridade superior.
	Um sistema embebido pode executar vários programas.
	Soft real time é quando o cumprimento das restrições causa redução da performance.
	Na plataforma arduino a função Serial.println(analogvalue, HEX), imprime um ASCII no formato octal
	No método de programação assíncrona, modificação nos requisitos implica modificações na estrutura do programa.
	Uma Task não é proprietária de recursos.
	Em programação tempo real podemos ter comunicação síncrona e comunicação assíncrona.
	Escalonamento Estático pode ser usado quando se está perante um sistema com comportamento determinístico.
	Uma Task possuí 4 estados: a correr, pronto a correr, morta, suspensa.

Grupo 2 (15 valores)

1. **(1,5 valor)** Indique quais são as principais caraterísticas de um sistema embebido.
2. **(1,5 valor)** Quais os desafios no design de sistemas embebidos.

3. **(1 valor)** O tempo de resposta a um deadline de sistema embebido de tempo real pode ser de dois tipos, descreva-os de acordo com a imagem abaixo.



4. **(1,5 valor)** Um cliente pretende fazer um sistema para gerir um parque de estacionamento. Entrega a informação que o parque terá capacidade para 100 veículos, uma entrada e uma saída. Elabore uma proposta de requisitos funcionais e não-funcionais.

5. **(1 valor)** Quais as vantagens e desvantagens da programação síncrona e assíncrona no âmbito dos sistemas embebidos.

6. **(1 valor)** Nos sistemas operativos é usual a utilização de métodos de sincronização, para evitar deadlock e livelock. Descreva em que consiste um deadlock e um livelock.

7. **(2,5 valor)** Para os seguintes 2 tipos de escalonador (FIFO e Rate-Monotonic) descreva as suas caraterísticas e ordene temporalmente as tarefas para cada escalonador.

Tarefa	Tempo execução (ms)	Período (ms)	Prioridade
A	10	40	1
B	20	60	2
C	10	80	3
D	20	100	4
E	20	120	5

8. **(2,5 valor)** Observe o seguinte código em Arduino e explique o seu objetivo geral. Descreva igualmente o funcionamento e objetivo de cada instrução.

```
int Pin = 9;
int cnt = 0;

void setup() {
  Serial.begin(57600);
  pinMode(8, INPUT);
}

void loop()
{
  if(digitalRead(8) == LOW)
  {
    Serial.print("SETR: ");
    Serial.print(cnt);
    Serial.print(" + ");
    Serial.println(analogRead(A3));
  }
```

```
  for (int Value = 0 ; Value <= 255; Value++)
  {
    analogWrite(Pin, Value);
    delay(5);
  }

  for (int Value = 255 ; Value >= 0; Value--)
  {
    analogWrite(Pin, Value);
    delay(5);
  }

  cnt++;
}
```

9. (2,5 valor) Analise o código FreeRTOS abaixo e descreva o objetivo de cada função.

```
SemaphoreHandle_t xSerialSemaphore;

void TaskDigitalRead( void *pvParameters );
void TaskAnalogRead( void *pvParameters );

void setup() {
  Serial.begin(9600);

  if ( xSerialSemaphore == NULL )
  {
    xSerialSemaphore = xSemaphoreCreateMutex();
    if ( ( xSerialSemaphore ) != NULL )
      xSemaphoreGive( ( xSerialSemaphore ) );
  }

  xTaskCreate(
    TaskDigitalRead
    ,   (const portCHAR *)"DigitalRead"
    ,   164
    ,   NULL
    ,   2
    ,   NULL );

  xTaskCreate(
    TaskAnalogRead
    ,   (const portCHAR *) "AnalogRead"
    ,   128
    ,   NULL
    ,   1
    ,   NULL );
}

void loop(){ }
```

```
/*--------------------- Tasks ---------------------*/

void TaskDigitalRead( void *pvParameters __attribute__((unused)) )
{
  uint8_t pushButton = 2;
  pinMode(pushButton, INPUT);

  for (;;)
  {
    int buttonState = digitalRead(pushButton);
    if ( xSemaphoreTake( xSerialSemaphore, ( TickType_t ) 5 ) == pdTRUE )
    {
      Serial.println(buttonState);
      xSemaphoreGive( xSerialSemaphore );
    }
    vTaskDelay(1);
  }
}

void TaskAnalogRead( void *pvParameters __attribute__((unused)) )
{
  for (;;)
  {
    int sensorValue = analogRead(A0);
    if ( xSemaphoreTake( xSerialSemaphore, ( TickType_t ) 5 ) == pdTRUE )
    {
      Serial.println(sensorValue);
      xSemaphoreGive( xSerialSemaphore );
    }
    vTaskDelay(1);
  }
}
```