

NOM : A B A B O U
PRENOM: S A R A H
INE: 1 7 0 0 2 5 8 8



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2, 0x12ABCD
addi r3, zero, 0x8FFF
stb r2, 0x6(zero)
ldh r4, 0x2(zero)
srai r5, r3, 8
```

	00	01	10	11
0x000	0xF1	0x23	0xC3	0x84
0x004	0x40	0x53	0x9D	0x62

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

r2	✓
r3	
r4	on
r5	

	00	01	10	11
0x000		✓		
0x004				

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

(Signature)

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

Z

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

peut

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

.data		Ident	valeur	0x0	0x1	0x2	0x3
<u>_tab:</u>		<u>_tab</u>		<i>a</i>			
<u>_root:</u>		<u>_root</u>					
<u>_current:</u>		<u>_current</u>				<i>X</i>	

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;

b

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;

b a b

NOM : M O R I N
PRENOM: T H I B A U D
INE: 20110805



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2,0x12ABCD
addi r3,zero,0x8FFF
stb r2,0x6(zero)
ldh r4,0x2(zero)
srai r5,r3,8
```

	00	01	10	11
0x000	0xF1	0x23	0xC3	0x84
0x004	0x40	0x53	0x9D	0x62

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

r2	<i>✓</i>
r3	<i>✓</i>
r4	<i>✓</i>
r5	

	00	01	10	11
0x000	<i>✓</i>	<i>✓</i>		
0x004				<i>✓</i>

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

oui

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4,0x1(zero)** ? Expliquez pourquoi.

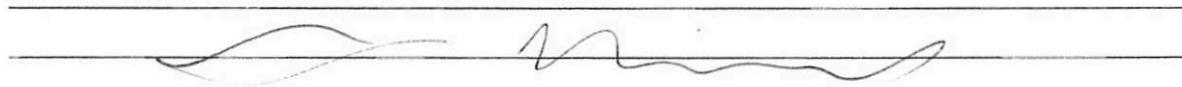
n

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).



Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

.data		Ident	valeur	0x0	0x1	0x2	0x3
<u>_tab:</u>		<u>_tab</u>					X
<u>_root:</u>	On	<u>_root</u>					
<u>_current:</u>		<u>_current</u>					
				0x000			
				0x004			
				0x008			
				0x00C			
				0x010			
				0x014			

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;	
	<i>(Signature)</i>

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;	
	<i>(Signature)</i>

NOM : G R A L L
PRENOM: J E S S Y
INE: 20200506



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2, 0x12ABCD
addi r3, zero, 0x8FFF
stb r2, 0x6(zero)
ldh r4, 0x2(zero)
srai r5, r3, 8
```

	00	01	10	11
0x000	0xF1	0x23	0xC3	0x84
0x004	0x40	0x53	0x9D	0x62

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

r2	✗
r3	
r4	✗
r5	

	00	01	10	11
0x000	✗			✗
0x004		✗		

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

ou

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

ou

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

tab

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

.data		Ident	valeur	0x0	0x1	0x2	0x3
<u>_tab:</u>	<i>or</i>	<u>_tab</u>		0x000		<i>x</i>	
<u>_root:</u>		<u>_root</u>		0x004			
<u>_current:</u>		<u>_current</u>		0x008		<i>x</i>	
				0x00C			<i>x</i>
				0x010			
				0x014	<i>x</i>		

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;

oui

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;

u

NOM : G U E R I N
PRENOM: A L E X Y S
INE: 170100078



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2, 0x12ABCD
addi r3, zero, 0x8FFF
stb r2, 0x6(zero)
ldh r4, 0x2(zero)
srai r5, r3, 8
```

	00	01	10	11
0x000	0xF1	0x23	0xC3	0x84
0x004	0x40	0x53	0x9D	0x62

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

r2	23
r3	42
r4	18
r5	12

	00	01	10	11
0x000		2		
0x004				2

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

(Handwritten answer: pas de

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

(Handwritten answer: r4 = 0x00000000)

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

On.

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

.data		Ident	valeur	0x0	0x1	0x2	0x3
_tab:	On	_tab	x				
_root:	On	_root	x		x		
_current:	On	_current	x			x	
				0x000			
				0x004		x	
				0x008			x
				0x00C			
				0x010			x
				0x014			

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;	On.
---------------	-----

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;	On
----------------------	----

NOM : H P I B A L L A
PRENOM: S E L M O U
INE: 2 0 2 0 9 3 9



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2,0x12ABCD
addi r3,zero,0x8FFF
stb r2,0x6(zero)
ldh r4,0x2(zero)
srai r5,r3,8
```

	00	01	10	11
0x000	0xF1	0x23	0xC3	0x84
0x004	0x40	0x53	0x9D	0x62

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

r2	x
r3	x
r4	ou
r5	n

	00	01	10	11
0x000		x		
0x004				x

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

Non

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4,0x1(zero)** ? Expliquez pourquoi.

Le contenu de l'adresse 0x1 est lu et placé dans r4.

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

b6b

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

.data		Ident	valeur	0x0	0x1	0x2	0x3
_tab:	23	_tab					X
_root:	42	_root					
_current:	18	_current				X	
				0x000			
				0x004			
				0x008			X
				0x00C		X	
				0x010	X		
				0x014			

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;	0000000000000000
---------------	------------------

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;	0000000000000000
----------------------	------------------

NOM : L A B R I G U I
PRENOM: S A C A H E D O I N E
INE: 20201784



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2, 0x12ABCD
addi r3, zero, 0x8FFF
stb r2, 0x6(zero)
ldh r4, 0x2(zero)
srai r5, r3, 8
```

	00	01	10	11
0x000	0xF1	0x23	0xC3	0x84
0x004	0x40	0x53	0x9D	0x62

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

r2	<u> </u>
r3	<u> </u>
r4	<u> </u>
r5	<u> </u>

	00	01	10	11
0x000		<u> </u>		
0x004				<u> </u>

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

non

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

0x1000 0x1004 0x1008 0x100C 0x1010 0x1014
0x1000 0x1004 0x1008 0x100C 0x1010 0x1014

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

.data		Ident	valeur	0x0	0x1	0x2	0x3
_tab:	fin	_tab	✓		0		
_root:	fin	_root	✓				
_current:	fin	_current				✓	

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;

fin

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;

fin

NOM : A K I L
PRENOM: R O C H I D I O M A R
INE: 2 0 2 0 1 8 3 8



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2,0x12ABCD
addi r3,zero,0x8FFF
stb r2,0x6(zero)
ldh r4,0x2(zero)
srai r5,r3,8
```

	00	01	10	11
0x000	0xF1	0x23	0xC3	0x84
0x004	0x40	0x53	0x9D	0x62

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

r2	0
r3	0
r4	0
r5	0

	00	01	10	11
0x000	X			
0x004				X

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

—

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4,0x1(zero)** ? Expliquez pourquoi.

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

Oui

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

.data		Ident	valeur	0x0	0x1	0x2	0x3
_tab:	X	_tab					
_root:	O	_root	X				
_current:	Z	_current					
		0x000					
		0x004					
		0x008					
		0x00C				1	
		0x010					
		0x014					

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;

O O O

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;

1 1 1

NOM : A K O T O
PRENOM: Y A O A R N A U D
INE: 2 0 1 1 1 6 4 6



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2,0x12ABCD
addi r3,zero,0x8FFF
stb r2,0x6(zero)
ldh r4,0x2(zero)
srai r5,r3,8
```

	00	01	10	11
0x000	0xF1	0x23	0xC3	0x84
0x004	0x40	0x53	0x9D	0x62

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

r2	nn
r3	oui
r4	n~
r5	

	00	01	10	11
0x000		✓		
0x004			✓	

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

non

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

oui.

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

on
on

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

.data		Ident	valeur	0x0	0x1	0x2	0x3
_tab:	O	_tab	X			X	
_root:	D	_root	X				
_current:	Z	_current					X

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;	test
---------------	------

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;	<i>on</i>
----------------------	-----------

NOM : M A
PRENOM: Q I A N
INE: 16011765



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2,0x12ABCD
addi r3,zero,0x8FFF
stb r2,0x6(zero)
ldh r4,0x2(zero)
srai r5,r3,8
```

	00	01	10	11
0x000	0xF1	0x23	0xC3	0x84
0x004	0x40	0x53	0x9D	0x62

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

r2	✓
r3	✓
r4	✓
r5	✓

	00	01	10	11
0x000		✓		
0x004			✓	

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

oui non

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

non 0, 0, 02

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

for

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

.data		Ident	valeur	0x0	0x1	0x2	0x3
<u>_tab:</u>	<i>m</i>	<u>_tab</u>	<i>X</i>				
<u>_root:</u>	<i>o</i>	<u>_root</u>	<i>0</i>				
<u>_current:</u>	<i>n</i>	<u>_current</u>	<i>1</i>				
				<i>X</i>			
				<i>X</i>			
				<i>X</i>			

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;	<i>on</i>
---------------	-----------

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;	<i>m</i>
----------------------	----------

NOM : N A V E T
PRENOM: B E N S A M I N
INE: 20213110



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2, 0x12ABCD
addi r3, zero, 0x8FFF
stb r2, 0x6(zero)
ldh r4, 0x2(zero)
srai r5, r3, 8
```

	00	01	10	11
0x000	0xF1	0x23	0xC3	0x84
0x004	0x40	0x53	0x9D	0x62

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

r2	1
r3	2
r4	3
r5	4

	00	01	10	11
0x000		X		
0x004			X	

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

oui

~~~~~

**Q3 :** Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

on

Adm

## Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

**Q1 :** Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

Node

**Q2 :** Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **\_tab**, **\_root**, **\_current** ainsi que le contenu de la mémoire au début du programme.

| .data     |     | Ident    | valeur | 0x0   | 0x1 | 0x2 | 0x3 |
|-----------|-----|----------|--------|-------|-----|-----|-----|
| _tab:     | oui | _tab     | ✓      |       |     |     |     |
| _root:    | non | _root    |        |       | ✓   |     |     |
| _current: |     | _current |        |       |     | ✓   |     |
|           |     |          |        | 0x000 |     |     |     |
|           |     |          |        | 0x004 |     | ✓   |     |
|           |     |          |        | 0x008 |     |     | ✓   |
|           |     |          |        | 0x00C |     |     |     |
|           |     |          |        | 0x010 |     |     |     |
|           |     |          |        | 0x014 |     |     |     |

**Q3 :** Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;

oui

**Q4 :** Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **\_current** est codée sur 16 bits)

root.next = current;

non