## 1 Home Run (14 pts)

 (3 pts) On considère le pseudo-code ci-dessous. Transformez le pour mettre en avant les blocs de base et respecter les contraintes des microprocesseurs. Identifiez clairement le nombre de blocs de base alors obtenus. Rappel cjump <cond>, <iffrue>, <iffalse>.

## Réponse :

```
a \leftarrow 41
  b \leftarrow [x] # Memory access
3 c ← 1
L1: cjump c \neq 10, L7, L9
5 L2: jump L1
6 L3: cjump b > 0, L2, L6
7 L4: b \leftarrow d
8 L5: jump L2
• L6: cjump b < 0, L2, L5
 L7: x \leftarrow x+4
   c \leftarrow c+1
   d \leftarrow [x]
   cjump b \neq d, L4, L8
L8: cjump b = 0, L2, L3
  L9: a \leftarrow a + 1
       return # a and b are liveout
```

 (1 pts) Dessinez le graphe de flot de contrôle de l'exemple ci dessus, c'est à dire l'exemple sans vos modifications de la question (1)! Vous utiliserez ℓ<sub>i</sub> pour indiquer la ligne i.

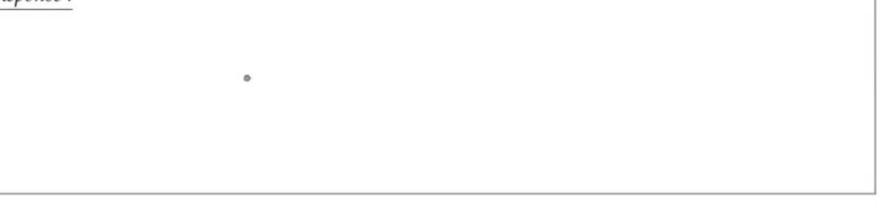
Réponse :		

3. **(4 pts)** Pour chaque nœud, indiquez les variables qui sont définies, celles qui sont utilisées, celles live\_in et celles qui sont live\_out. Pour rappel :

$$in[n] = use[n] \cup (out[n] \setminus def[n])$$
  $out[n] = \bigcup_{s \in succ(n)} (in[s])$ 

éponse :				
de	ef l	ise	live_in	live_out
$\ell_1$				
$\ell_2$				
$\ell_3$ $\ell_4$				
$\ell_4$				
$\ell_5$				
$\ell_6$				
$\ell_7$				
$\ell_8$				
l9				
$\ell_{10}$				
$\ell_{11}$				
$\ell_{12}$				
$\ell_{13}$				
$\ell_{14}$				
$\ell_{15}$				
$\ell_{16}$				

	Re-dessinez le graphe de de chaque variable.	flot de contrôle	de la question	(2) en y faisant	apparaître la
Réponse :					



5. (1 pts) Déduisez en le graphe d'interférence du p	orogramme.	
Réponse :		

6. (1 pts) Calculez la spill priority des variables a, b, c, d et x. Pour rappel :  $spill\_priority = ((use\_outside\_loop + def\_outside\_loop) +$  $(10 * use\_inside\_loop + def\_inside\_loop))/degree$ Réponse :

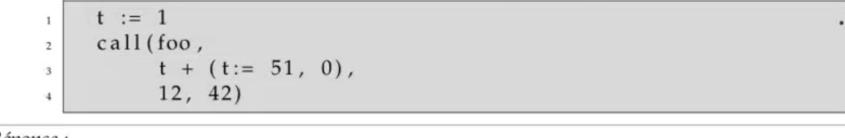
7. (2 pts) Réécrivez le code en effectuant un spill de a à l'adresse [sp+4]. Les lignes ci-dessous sont là pour vous aider à écrire droit, il peut y en avoir plus que nécessaire.
Réponse :

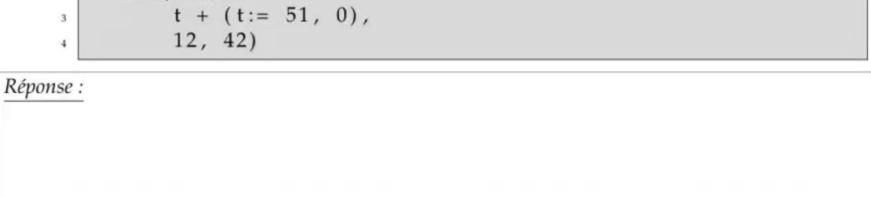
$\ell_1$	$ \ell_{11} $	
$\ell_2$	$\ell_{12}$	
$\ell_3$	$\ell_{13}$	
$\ell_4$	$\ell_{14}$	
$\ell_5$	$\ell_{15}$	
$\ell_6$	$\ell_{16}$	
$\ell_7$	$\ell_{17}$	
$\ell_8$	$\ell_{18}$	
<i>l</i> 9	$\ell_{11}$ $\ell_{12}$ $\ell_{13}$ $\ell_{14}$ $\ell_{15}$ $\ell_{16}$ $\ell_{17}$ $\ell_{18}$ $\ell_{19}$	
$\ell_{10}$	$\ell_{20}$	

(1 pts) En ignorant la variable a, que fait le programme de la question 1?			
éponse :			

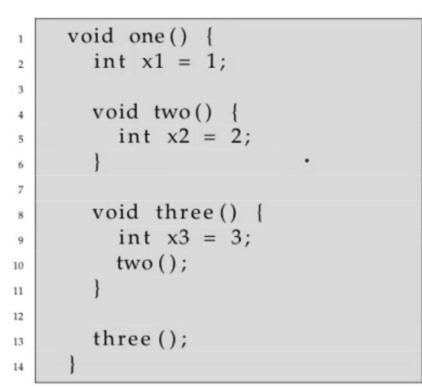
<ol> <li>(1 pt) À quoi sert le déroulage de boucle? Vous donnerez un exemple haut niveau (code C) pe expliquer votre propos.</li> </ol>					uı
		Page 3			
Réponse :					

3. (1 pts) Qu'est ce que la linéarisation? Linéarisez le code suivant.





4. (1 pt) Expliquez le rôle du middle-end dans un compilateur.				
Réponse :				



6. (1 pt) Quelles sont les trois traductions possibles pour l'expression α > β? Il ne vous est pas demandé du code mais uniquement l'explication des trois cas.
<u>Réponse</u>: