1. Que fait le programme suivant ?
2. Décrivez le code et son fonctionement en détail.
3. Que contiendra le fichier une fois le progamme terminé ? Pourquoi ?
4. Ici on utilise un fichier pour créer une mémoire partagée, comment modifier le code pour ne pas utiliser de fichier (i.e. uniquement une mémoire partagée)
5. Notre programme peu créer un processus orphelin. Pourquoi ? Pourrait-il créer un processus zombie ? Pourquoi ?
6. A quoi sert la fonction sched_yield() ?
7. Pourquoi utiliser la déclaration de type number_t plutôt qu'un unsigned int directement ?
8. Ce programme, plus particulièrement la fonction cons, peut bloquer dans un cas précis. Pouvez-vous l'identifier ?

exam_blanc.c

```c
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/mman.h>
#include <fcntl.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <sched.h>

#define SIZE 20000000
#define PLEASE_WAIT -1

typedef unsigned int number_t;

void OnError(const char *str)
{
        perror(str);
        exit(EXIT_FAILURE);
}


int prod(number_t *data)
{
    int fd;
    if( (fd = open("/dev/urandom", O_RDONLY)) == -1)
        OnError("open 2");

    for(int i=0;i<SIZE;i++)
        if(read(fd, data+i, sizeof(number_t)) != sizeof(number_t))
            OnError("read");

    close(fd);
```

```
33 }
34
35
36 int cons(number_t *data)
37 {
38     for(int i=0;i<SIZE;i++) {
39         while(*(data+i) == PLEASE_WAIT)
40             sched_yield();
41         printf("%d: %x\n", i, *(data+i));
42     }
43 }
44
45
46 int main(int argc, char* argv[])
47 {
48     int fd, i;
49     number_t *data;
50
51     if(argc != 2)
52     {
53         printf("Usage: one filename\n\n");
54         exit(0);
55     }
56
57     size_t mem_size = sizeof(number_t)*SIZE;
58
59     if( (fd = open(argv[1], O_RDWR | O_CREAT, S_IRUSR | S_IWUSR)) == -1)
60         OnError("open");
61
62     if( ftruncate(fd, mem_size) == -1)
63         OnError("ftruncate");
64
65     data = mmap(NULL, mem_size, PROT_READ | PROT_WRITE, MAP_SHARED, fd,
        0);
66     if(data == MAP_FAILED)
67         OnError("mmap");
68
69     for(int i=0;i<SIZE;i++)
70         *(data+i) = PLEASE_WAIT;
71
72     pid_t res = fork();
73     if(res == 0)
74         cons(data);
75     else if(res > 0)
76         prod(data);
77     else
78         OnError("fork");
79
80     if(munmap(data, mem_size) == -1)
81         OnError("munmap");
82
83     close(fd);
84
85     return 0;
86 }
```

makefile

```
1 all: exam_blanc
2
3 exam_blanc: exam_blanc.c
4         gcc -g -std=gnu11 exam_blanc.c -o exam_blanc
```