

**SYSTEM D’EXPLOITATION**

**Mini projet**

**Réaliser par :**

**NADRANI OUSSAMA TAZI Achraf :**

1311778906 1513755449

**Encadré par:**

**Mr MEKNASSI**

* **Remerciement**
* Ce modeste travail que nous avons réalisé est le fruit des efforts fourni par Mr. MEKNASSI qui nous a poussés à donner le meilleur de nous-mêmes, et aussi grâce à sa pédagogie unique qui permet à l’étudiant de s’implique dans la construction de son cours, et nous permet ainsi d’élucidé toute ambiguïté concernant le module System d’exploitions qu’est le cœur de la formation d’un informaticien Nous tenons à vous remercier Mr. MEKNASSI pour vos efforts et vos précieux conseils qui nous permettent de développer nos compétences.
* **Plan**
* **Énoncé :**
* **Programme :**
* **Structure stockant les informations des threads clients et du magasin .**
* **Fonction pour tirer un nombre au sort entre 0 et max .**
* **Fonction pour le thread du stock hopital .**
* **Fonction pour les threads des clients Medcin .**
* **Fonctin principale .**
* **Programme generale .**
* **Exécution :**

* **Énoncé**
* Ecrire un programme en C qui en utilisant les threads, gère le stock d’un hôpital qui est composé des produits suivants: pansements, ciseaux et antiseptiques. Dans ce programme vous créez trois threads dont chacun gère un de ces produits et dix threads pour la gestion des médecins qui veulent se procurer de ces produits médicamenteux.
* Les threads médecins prennent dans le stock et les threads du magasin vont réapprovisionner le stock dès qu'il devient trop bas pour satisfaire les clients (200 pansements, 60 ciseaux et 150 antiseptiques). Le nombre d'articles pris du stock sont des nombres aléatoires ainsi que l'ordre de passage des médecins.
* Le nombre d'articles pris du stock sont des nombres aléatoires qui ne dépassent pas une certaine valeur par demande à savoir : 85 pour les pansements, 05 pour les ciseaux et 15 pour l’antiseptique. L'ordre de passage des clients est aussi aléatoire.
* Au début du programme le stock contient les quantités suivantes: 1500 pansements, 250 ciseaux et 1100 antiseptiques.
* **Programme :**
* **Structure stockant les informations des threads clients et du magasin .**

**/\* Structure stockant les informations des threads clients et du magasin. \*/**

**typedef struct{ int stock\_pans; int stock\_cise; int stock\_anti;**

**pthread\_t thread\_store\_pans;**

**pthread\_t thread\_store\_cise;**

**pthread\_t thread\_store\_anti;**

**pthread\_t thread\_clients [NB\_MEDECIN];**

**pthread\_mutex\_t mutex\_stock\_pans;**

**pthread\_cond\_t cond\_stock\_pans;**

**pthread\_cond\_t cond\_clients\_pans;**

**pthread\_mutex\_t mutex\_stock\_cise;**

**pthread\_cond\_t cond\_stock\_cise;**

**pthread\_cond\_t cond\_clients\_cise;**

**pthread\_cond\_t cond\_clients\_cise;**

**pthread\_mutex\_t mutex\_stock\_anti;**

**pthread\_cond\_t cond\_stock\_anti;**

**pthread\_cond\_t cond\_clients\_anti; }**

* **Fonction pour tirer un nombre au sort entre 0 et max .**

/\* Fonction pour tirer un nombre au sort entre 0 et max. \*/

static int get\_random (int max)

{

double val;

val = (double) max \* rand ();

val = val / (RAND\_MAX + 1.0);

return ((int) val);

}

* **Fonction pour le thread du stock hopital .**

/\* Fonction pour le thread du magasin. \*/

static void \* fn\_store\_pans (void \* p\_data)

{

while (1)

/\* Debut de la zone protegee. \*/

pthread\_mutex\_lock (& store.mutex\_stock\_pans);

pthread\_cond\_wait (& store.cond\_stock\_pans, & store.mutex\_stock\_pans);

store.stock\_pans = INITIAL\_STOCK\_pans;

printf ("Remplissage du stock de %d pansements !\n", store.stock\_pans);

pthread\_cond\_signal (& store.cond\_clients\_pans);

pthread\_mutex\_unlock (& store.mutex\_stock\_pans);

/\* Fin de la zone protegee. \*/

}

return NULL;

* **Fonction pour les threads des clients Medcin .**

**/\* Fonction pour les threads des clients. \*/**

**static void \* fn\_clients (void \* p\_data)**

**{**

**int nb = (int) p\_data;**

**while (1) {**

**int produit= get\_random(4);**

**int val;**

**if(produit==1)**

**{ val = get\_random (101);**

**psleep (get\_random (3));**

**/\* Debut de la zone protegee. \*/**

**pthread\_mutex\_lock (& store.mutex\_stock\_pans);**

**if (val > store.stock\_pans)**

**{**

**pthread\_cond\_signal (& store.cond\_stock\_pans);**

**pthread\_cond\_wait (& store.cond\_clients\_pans, & store.mutex\_stock\_pans);**

**}**

**store.stock\_pans = store.stock\_pans - val;**

**printf ("Client %d prend %d du pansements, reste %d en stock !\n " ,nb, val, store.stock\_pans );**

**pthread\_mutex\_unlock (& store.mutex\_stock\_pans);**

**/\* Fin de la zone protegee. \*/}**

**if(produit==2)**

**{ val = get\_random (11);**

**psleep (get\_random (3));**

**/\* Debut de la zone protegee. \*/**

**pthread\_mutex\_lock (& store.mutex\_stock\_cise);**

**if (val > store.stock\_cise)**

**{**

**pthread\_cond\_signal (& store.cond\_stock\_cise);**

**pthread\_cond\_wait (& store.cond\_clients\_cise, & store.mutex\_stock\_cise);**

**}**

**store.stock\_cise = store.stock\_cise - val;**

**printf ("Client %d prend %d du ciseaux, reste %d en stock !\n",nb, val, store.stock\_cise );**

* **Fonction principale .**

int main (void)

{

int i = 0;

int ret = 0;

/\* Creation des threads. \*/

printf ("Creation du thread du pasement !\n");

ret = pthread\_create (

& store.thread\_store\_pans, NULL,

fn\_store\_pans, NULL );

printf ("Creation du thread du ciseaux !\n");

ret = pthread\_create (& store.thread\_store\_cise, NULL, fn\_store\_cise, NULL);

printf ("Creation du thread du antiseptique !\n");

ret = pthread\_create (& store.thread\_store\_anti, NULL,fn\_store\_anti, NULL);

/\* Creation des threads des clients si celui du magasinn a reussi. \*/

if (! ret) {

printf ("Creation des threads clients !\n");

for (i = 0; i < NB\_MEDECIN; i++) {

ret = pthread\_create ( & store.thread\_clients [i], NULL,fn\_clients, (void \*) i);

if (ret) {

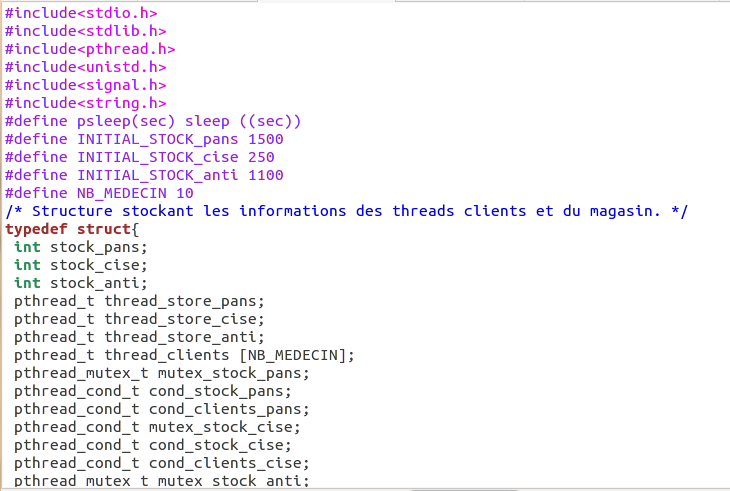
fprintf (stderr, "%s",strerror(ret));

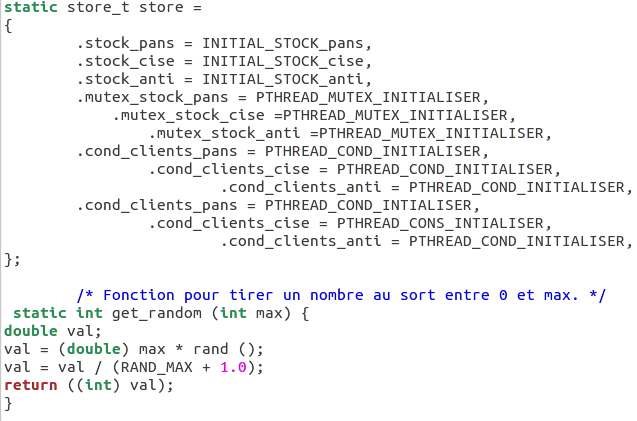
}

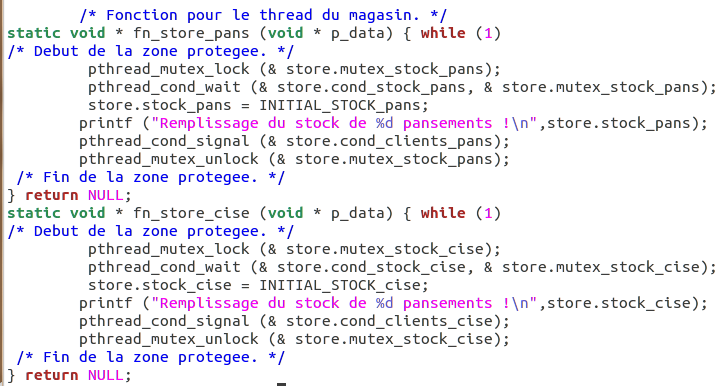
}

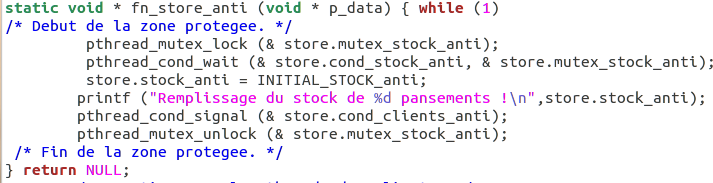
}

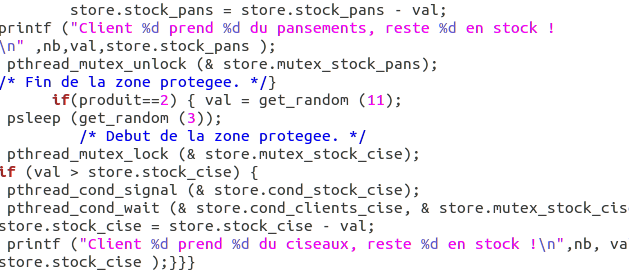
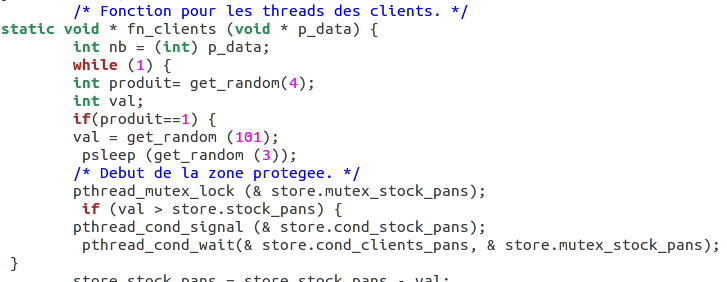
* **Programme generale .**

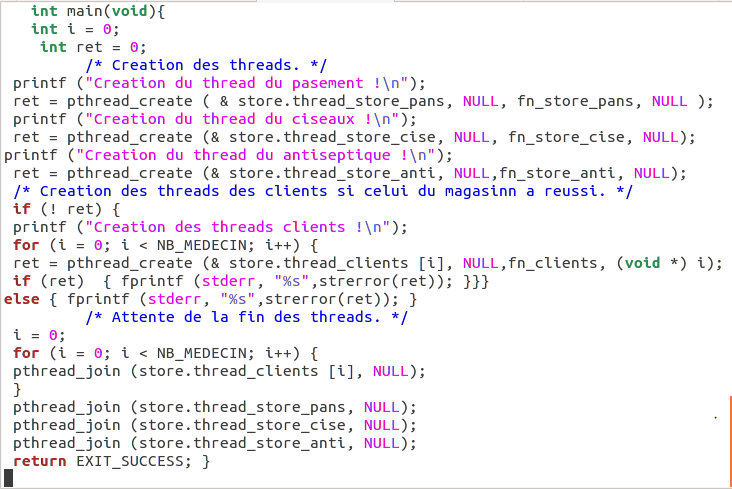












* **Exécution :**