Esame Sistemi Operativi - Operating Systems Exam
2022/06/14

Ex 1 (5.0 points)

Italiano

Un file di tipo ASCII immagazzina una sequenza di record. Ogni record include 3 campi: un valore intero, una stringa e un numero reale. I tre campi hanno tutti una dimensione variabile e sono separati da un numero variabile di spazi. Il seguente è un esempio di file con tale formato:

```
15345 Acceptable 26.50
146467 Average 23.75
. . .
```

Si scriva un tratto di codice C che, utilizzando le system calls UNIX **open(), read()**, e **close()** per effettuare l'I/O da file, memorizzi tali record in un array di strutture di tipo `record_t`, definito nel modo seguente:

```
#define N 100

typedef struct record_s {
  int i;
  char s[N];
  float f;
} record_t;
```

Suggerimento: Si ricordi che la system call **read()** manipola i file in formato binario.

English

An ASCII file stores a sequence of records. Each record includes 3 fields: an integer value, a string, and a real number. All three fields have variable size and are separated by a variable number of white spaces. The following is an example of such a file:

```
15345 Acceptable 26.50
146467 Average 23.75
. . .
```

Write a segment of C code that, using the UNIX system calls **open(), read()**, and **close()** to perform file I/O, stores these records into an array of structures of type `record_t`, defined as follows:

```
#define N 100

typedef struct record_s {
  int i;
  char s[N];
  float f;
} record_t;
```

Suggestion: Keep in mind that the system call read() manipulates files in binary form.

Risposta: Answer:

```
typedef struct record_s {
  int i;
  char s[N];
```

```
  float f;
  struct record_s *next;  // Added
} record_t;

char c;
int fd, not_end1;
char line[MAXIMUM_LINE_LENGTH];
int position = 0;
record_t *r, *head;

// iterate till the end of file
head = NULL;
do {
  position = 0;
  do {
    not_end = read (fd, &c, sizeof(char));
    if (not_end != 0) {
      line[position] = c;
      position++;
    }
    // I suppose that newline are represented only by a '/n'
  } while (c!='\n' && not_end);

  r = (record_t *) malloc (1 * sizeof (record_t));
  sscanf (line, "%d %s %f", &r->i, r->s, &r->f);
  r->next = head;
  head = r;
} while (not_end);
```

Ex 2 (3.0 points)

Italiano

Si supponga che il seguente programma venga eseguito con il valore 3 sulla linea di comando. Si indichino quanti caratteri 'X' sono stampati dal programma su standard output. Indicare nella risposta solo un numero intero.

English

Suppose that the following program is executed passing the value 3 on the command line. Report the number of characters 'X' printed by the program on standard output. Indicate in the response only an integer number.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main (int argc, char *argv[]) {
  char str[100];
  int n = atoi(argv[1]);

  setbuf(stdout, 0);
  if (n>0) {
    fork();
    if (fork()) {
      printf("X");
      sprintf(str, "%s %d", argv[0], n-1);
      system(str);
    }
  }
```

```
  return (0);
}
```

Risposta: Answer:
14

Ex 3 (2.5 points)

Italiano

Si supponga che il disco rigido di un piccolo sistema embedded sia costituito da 26 blocchi di 1 MByte, che tali blocchi siano numerati da 0 a 25, che il sistema operativo mantenga traccia dei blocchi liberi (occupati) indicandoli in un vettore con il valore 0 (1), e che la situazione attuale del disco sia rappresentata dal seguente vettore:

`0 1 1 0 0 0 1 0 0 0 1 1 1 1 1 0 0 0 0 1 0 0 1 0 0 1`

Si indichino quali delle seguenti affermazioni sono vere. Si osservi che risposte errate implicano una penalità sul punteggio finale.

English

Suppose that the hard disk of a small embedded system is composed of 26 blocks of 1 MByte each, which are numbered from 0 to 25. Suppose that the operating system keeps track of the free (occupied) blocks indicating them in a vector with the value 0 (1), and that the current situation of the disk is represented by the following vector:

`0 1 1 0 0 0 1 0 0 0 1 1 1 1 1 0 0 0 0 1 0 0 1 0 0 1`

Indicate which of the following statements are correct. Note that wrong answers imply a penalty on the final score.

Scegli una o più alternative: Choose one or more options:

1. ✅ Con la strategia di allocazione concatenata possono essere allocati un file di dimensione 4.8 MByte e un file di dimensione 5.3 MByte. With the linked allocation strategy a file of dimension 4.8 MByte and a file of dimension 5.3 MByte can be allocated.
2. ✅ Con la strategia di allocazione indicizzata può essere allocato un file di dimensione 10.8 MByte. With the indexed allocation strategy a file of dimension 10.8 MByte can be allocated.
3. ⬜ Con la strategia di allocazione contigua ottenuta mediante l'algoritmo WORST-FIT possono essere allocati nell'ordine i file F1 di 1.7 MByte, F2 di 0.2 MByte e F3 di 3.1 MByte. With the contiguous allocation strategy based on the WORST-FIT algorithm, the following files can be allocated in the following order: F1 of 1.7 MByte, F2 of 0.2 MByte, and F3 of 3.1 Mbyte.
4. ✅ La strategia di allocazione concatenata non soffre di frammentazione esterna. The linked allocation strategy does not suffer from external fragmentation.
5. ⬜ Con la strategia di allocazione contigua ottenuta mediante l'algoritmo FIRST-FIT possono essere allocati nell'ordine i file F1 di 1.3 MByte, F2 di 2.1 MByte, F3 di 1.2 MBytes, e F4 di 3.4 MByte. With the contiguous allocation strategy based on the FIRST-FIT algorithm, the following files can be allocated in the following order: F1 of 1.3 MByte, F2 of 2.1 MByte, F3 of 1.2 MBytes, and F4 of 3.4 Mbyte.
6. ✅ Con la strategia di allocazione contigua ottenuta mediante l'algoritmo BEST-FIT possono essere allocati nell'ordine i file F1 di 1.3 MByte, F2 di 2.1 MByte, F3 di 1.2 MBytes, e F4 di 3.4 MByte. With the contiguous allocation strategy based on the BEST-FIT algorithm, the following files can be allocated in the following order: F1 of 1.3 MByte, F2 of 2.1 MByte, F3 of 1.2 MBytes, and F4 of 3.4 Mbyte.

Ex 4 (2.0 points)

Italiano

Relativamente alle system call **stat()**, **lstat()** e **fstat()**, si indichino quali delle seguenti affermazioni sono corrette. Si osservi che risposte errate implicano una penalità nel punteggio finale.

English

Regarding the system calls **stat()**, **lstat()**, and **fstat()**, indicate which of the following statements are correct. Note that incorrect answers imply a penalty in the final score.

Scegli una o più alternative: Choose one or more options:

1. ⬜ Nel caso in cui ci sia un link simbolico `link_f1` che punta al file `f1`, la system call lstat() ritorna informazioni relativamente al file `f1`. If there is a symbolic link `link_f1` pointing to file `f1`, the lstat() system call returns information about file `f1`.
2. ✅ Le system call stat(), lstat() e fstat() sono tutte caratterizzate dagli stessi valori di ritorno. The system calls stat(), lstat(), and fstat() are all characterized by the same return values.
3. ✅ Tra le informazioni che possono essere ottenute dalle system call stat(), lstat() e fstat() vi è il numero di i-node relativo al file su cui la system call è stata eseguita. Among the pieces of information that can be obtained from the system calls stat(), lstat() and fstat() there is the i-node number associated with the file on which the system call has been executed.
4. ✅ Le system call stat() e fstat(), se applicate su un file di tipo regolare (cioè non un link simbolico), sono equivalenti. The system calls stat() and fstat(), if applied on a regular file (i.e., not a symbolic link), are equivalent.
5. ⬜ Le tre system call permettono di elencare le entries contenute in una directory. The three system calls allow us to list the entries contained in a directory.
6. ✅ Le tre system call restituiscono un puntatore ad una struttura di tipo `struct stat`. The three system calls return a pointer to a structure of type `struct stat`.

Ex 5 (2.0 points)

Italiano

Si consideri il caso in cui un processo diventi “orfano”. Si indichi quali delle seguenti affermazioni sono corrette. Si osservi che risposte errate implicano una penalità nel punteggio finale.

English

Consider the case in which a process becomes “orphan”, indicate which of the following statements are correct. Note that incorrect answers imply a penalty in the final score.

Scegli una o più alternative: Choose one or more options:

1. ⬜ Esso attende che il padre effettui una system call wait() al fine di rilasciare tutte le sue risorse. It waits until the parent executes a wait() system call in order to release all its resources.
2. ⬜ Quando il processo effettuerà la system call wait() non sarà più orfano. When the process will perform the system call wait() it will no longer be an orphan.
3. ⬜ Il processo verrà ereditato dal terminale/shell che lo aveva eseguito. The process will be inherited by the terminal/shell that executed it.
4. ✅ Il processo verrà ereditato da un altro processo. The process will be inherited by another process.
5. ⬜ Il processo non ha nessun padre. The process has no parent.
6. ✅ Il processo può ancora entrare nello stato di “esecuzione”. The process can still enter the "running" state.

Ex 6 (2.0 points)

Italiano

Il seguente tratto di codice mostra una possibile implementazione della system call **alarm()** mediante altre system call. Si suppone il codice debba attivare l'allarme dopo N secondi. Si indichi quali delle seguenti affermazioni sono corrette. Si osservi che risposte errate implicano una penalità nel punteggio finale.

English

The following piece of code shows a possible implementation of the **alarm()** system call by means of other system calls. Suppose that the code triggers the alarm after N seconds. Please, indicate which of the following statements are correct. Note that incorrect answers imply a penalty in the final score.

```
void myAlarm (int sig) {
  if (sig==SIGUSR1)
    printf ("Alarm on ...\n");
  return;
}

...
pid_t pid;
signal (SIGUSR1, myAlarm);
...
if (fork()) {
  pause (N);
  kill (getppid(), SIGUSR1);
  exit (0);
}
...
```

Scegli una o più alternative: Choose one or more options:

1. ⬜ Nel tratto di codice indicato il segnale viene mandato dal padre al figlio. In the previous piece of code the signal is sent from the parent to the child.
2. ✅ La system call kill() non è usata correttamente perché occorre inviare un segnale al processo figlio non al processo padre. The kill() system call is not used correctly because a signal must be sent to the child process, and not the parent process.
3. ✅ Il processo deve utilizzare la system call sleep() non la pause(). The process must use the system call sleep() and not the pause().
4. ⬜ Occorre utilizzare il segnale SIGALRM, non SIGUSR1. The signal SIGALRM has to be used, and not SIGUSR1.
5. ⬜ La system call fork() può essere sostituita dalla exec(). The fork() system call can be replaced by the exec().

Ex 7 (5.0 points)

Italiano

Un numero imprecisato di thread può invocare una delle seguenti due funzioni:

1. `void wait_ch(int x)`, con `0<=x<10`. Questa funzione blocca il thread chiamante finché non viene invocata la funzione `signal_ch(int y)`, con `y==x`.
2. `void signal_ch(int y)`, con `0<=y<10`. Questa funzione risveglia tutti i thread che hanno invocato in precedenza la funzione `wait_ch(int x)`, con `x==y`. La funzione non ha effetto se nessun thread soddisfa la condizione enunciata al momento in cui essa viene chiamata.

Esempio di funzionamento:

- Il thread `A` chiama `wait_ch(4)` e viene posto in attesa.
- Il thread `B` chiama `wait_ch(2)` e viene posto in attesa.
- Il thread C chiama `signal_ch(8)` senza alcun effetto.
- Il thread `D` chiama `wait_ch(4)` e viene posto in attesa.
- Il thread E chiama `signal_ch(4)` risvegliando i thread `A` e `D`.
- Il thread D chiama `signal_ch(2)` risvegliando il thread `B`.

Si realizzino le funzioni `wait_ch()` e `signal_ch()` usando i semafori POSIX, definendone le relative variabili condivise.

English

An unknown number of threads can invoke one of the following two functions:

1. `void wait_ch(int x)`, with `0<=x<10`. This function blocks the calling thread until the function `signal_ch(int y)` is called with `y==x`.
2. `void signal_ch(int y)`, with `0<=y<10`. This function wakes up all the threads that previously called the function `wait_ch(int x)` with `x==y`. The function has no effect if no thread satisfies the condition previously described at the time it is called.

Example of operation:

- Thread `A` calls `wait_ch(4)` and it is blocked.
- Thread `B` calls `wait_ch(2)` and it is blocked.
- Thread C calls `signal_ch(8)` without any effect.
- Thread `D` calls `wait_ch(4)` and it is blocked.
- Thread E calls `signal_ch(4)` that wakes up threads `A` and `D`.
- Thread D calls `signal_ch(2)` that wakes up thread `B`.

Implement the functions `wait_ch()` and `signal_ch()` using POSIX semaphores, defining the related shared variables.

Risposta: Answer:

```
// Solution 1 (CORRECT WITH THE ALLOWED HYPOTHESIS OF ACYCLIC THREADS)
#define MAX_WQ 10

int n_wait[MAX_WQ];
sem_t sync[MAX_WQ];
sem_t m[MAX_WQ];

int main(){
  for (int i=0; i<MAX_WQ; i++){
    n_wait[i] = 0;
    sem_init(&sync[i], 0, 0);
    sem_init(&m[i], 0, 1);
  }

  ...
}

void wait_ch(int x){
  sem_wait(&m[x]);
  n_wait[x]++;
  sem_post(&m[x]);

  sem_wait(&sync[x]);
}

void signal_ch(int y){
  sem_wait(&m[y]);
  for (int i=0; i<n_wait[y]; i++){
    sem_post(&sync[y]);
  }
  n_wait[y]=0;
  sem_post(&m[y]);
}


// Solution 2 (CORRECT WITH THE ALLOWED HYPOTHESIS OF CYCLIC THREADS)
```

```
#define MAX_WQ 10

int n_wait[MAX_WQ];
sem_t sync[MAX_WQ];
sem_t m[MAX_WQ];
sem_t b[MAX_WQ];

int main(){
  for (int i=0; i<MAX_WQ; i++){
    n_wait[i] = 0;
    sem_init(&sync[i], 0, 0);
    sem_init(&m[i], 0, 1);
    sem_init(&b[i], 0, 1);
  }

  ...
}

void wait_ch(int x){
  sem_wait(&b[x]);
  sem_wait(&m[x]);
  n_wait[x]++;
  sem_post(&m[x]);
  sem_post(&b[x]);

  sem_wait(&sync[x]);

  sem_wait(&m[x]);
  n_wait[x]--;
  if (n_wait[x]==0)
    sem_post(&b[x]);
  sem_post(&m[x]);
}

void signal_ch(int y){
  sem_wait(&b[y]);
  sem_wait(&m[y]);
  for (int i=0; i<n_wait[y]; i++){
    sem_post(&sync[y]);
  }
  sem_post(&m[y]);
}
```

Ex 8 (3.0 points)

Italiano

Si indichi quali delle seguenti affermazioni riferite ai comandi bash sono corrette. Si osservi che risposte errate implicano una penalità nel punteggio finale.

English

Indicate which of the following statements related to bash commands are correct. Note that incorrect answers imply a penalty in the final score.

Scegli una o più alternative: Choose one or more options:

1. ⬜ Il commando: The command:
   `echo "three" > g; echo -e "four\nfive\nsix" > g; cat g | wc -l`

fornisce il seguente output: Provides the following output:
```
1
```

2. ✅ Il commando: The command:
```
echo "three" > g; echo "four\nfive\nsix" >> g; cat g | wc -l
```
fornisce il seguente output: Provides the following output:
```
2
```

3. ⬜ Il commando: The command:
```
echo "abaaab aaxxxaaaaaabbbaaas abbbeeedaaaa"|egrep -e
"aax{3}aaa(aaa)*bbb+aaas"; echo $?
```
fornisce il seguente output: Provides the following output:
```
1
```

4. ✅ Il commando: The command:
```
echo "abaaab aaxxxaaaaaabbbaaas abbbeeedaaaa"|tr -d a|egrep -e "xxxa*b+s";
echo $?
```
fornisce il seguente output: Provides the following output:
```
bb xxxbbbs bbbeeed
0
```

5. ✅ Il commando: The command:
```
echo "stefano 10 aaa x" > f; echo "giulia 5 aaa x" >> f; echo "lodovica 14 bbb
x" >> f; echo "gabriele 13 aaa x" >> f; cat f | cut -d " " -f 2,4|sort -nr
```
fornisce il seguente output: Provides the following output:
```
14 x
13 x
10 x
5 x
```

6. ⬜ Il commando: The command:
```
echo "stefano 10 aaa x" > f; echo "giulia 5 aaa x" >> f; echo "lodovica 14 bbb
x" >> f; echo "gabriele 13 aaa x" >> f; cat f | egrep -v bbb|head -n 1
```
fornisce il seguente output: Provides the following output:
```
lodovica 14 bbb x
```

7. ✅ Il commando: The command:
```
echo "stefano 10 aaa x" > f; echo "giulia 5 aaa x" >> f; echo "lodovica 14 bbb
x" >> f; echo "gabriele 13 aaa x" >> f; head -n 3 f| tail -n 1|cut -d " " -f 3
```
fornisce il seguente output: Provides the following output:
```
bbb
```

8. ⬜ Il commando: The command:
```
echo "stefano 10 aaa x" | tr sa1e xb2x| tr e y
```
fornisce il seguente output: Provides the following output:q
```
xtyfbno 20 bbb x
```

Ex 9 (3.0 points)

Italiano

Si consideri il seguente insieme di processi. Rappresentare mediante diagramma di Gantt l'esecuzione di tali processi utilizzando l'algoritmo Priority Scheduling (PS), al fine di calcolare il tempo di terminazione di ciascun processo e il tempo di attesa medio.

Si prega di riportare la risposta su un'unica riga, indicando i tempi di terminazione di P1, P2, P3, P4 e P5 seguiti dal tempo di attesa medio. Separare i numeri con un unico spazio. Riportare il tempo di attesa medio con 1 sola cifra decimale. Non inserire nessun altro carattere nella risposta. Esempio di risposta corretta: 20 23 11 45 67 30.5

English

Consider the following set of processes. Represent using a Gantt diagram the execution of these processes using the Priority Scheduling (PS) scheduling algorithm, in order to compute the termination time of each process, and the average waiting time.

Please, write your answer on a single line, indicating the termination times of P1, P2, P3, P4 and P5 followed by the average waiting time. Separate the numbers with a single space. Report the average waiting time with a

single decimal digit. Do not enter any other character in the response. Example of correct answer: 20 23 11 45 67 30.5

Processo Process	TempoArrivo ArrivalTime	BurstTime	Priorità Priority
P1	0	15	1
P2	2	21	3
P3	4	17	5
P4	6	13	4
P5	8	15	2

Soluzione. Solution.
P1: start=0, end=14, waitingTime=0
P5: start=15, end=29, waitingTime=7
P2: start=30, end=50, waitingTime=28
P4: start=51, end=63, waitingTime=45
P3: start=64, end=80, waitingTime=60
AverageWaitingTime: 28.00
GanttChart:
1111111111111115555555555555522222222222222222222244444444444443333333333333333

Risposta. Answer.
15 51 81 64 30 28.0

Ex 10 (5.0 points)

Italiano

Realizzare uno script bash dal nome **clean.sh** che, ricevuto il nome di un direttorio sulla riga di comando, ricerca e cancella all'interno di tale albero di direttori tutti i file che iniziano con "." o che terminano con "~", ma solo nel caso in cui il numero di caratteri che compongono il nome del file sia pari. Se il numero di caratteri che compongono il nome del file è dispari deve aggiungere in fondo al nome del file l'estensione ".odd". Ad esempio il file "hi~" deve essere rinominato in "hi~.odd".

English

Implement a bash script with name **clean.sh** that, given in input the name of a directory through command line, searches and deletes in such a directory tree all files that begin with a "." or that end with a "~", but only in the case the number of characters composing the name of the file is even. If the number of characters that compose the name of the file is odd, the script must add at the end of the file name the extension ".odd". For instance, the file "hi~" has to be renamed to "hi~.odd".

Risposta: Answer:

```
#!/usr/bin/env bash
################################################################################
# Version 1: Using find with -regex and ${#}
################################################################################

# Check if the correct parameters have been passed
if [ $# -lt 1 ]; then
  echo "Usage: ./clean.sh <dir>"
  exit 1
fi

# Find files
find $1 -type f -regextype posix-extended -regex ".*/(\..*)|(.*~)$" > tmp.txt

# Erase files with an even length number, rename those with an odd length name
while read line; do
```

```
  name=$(basename $line)
  n=${#name}
  let "even=n%2"
  if [ $even -eq 0 ]; then
    rm $line
  else
    mv $line $line".odd"
  fi
done < tmp.txt

# Clean temporary file
rm tmp.txt

exit 0


#!/usr/bin/env bash
################################################################################
# Version 2: Using find with two -name options in OR and wc
################################################################################

# Check if the correct parameters have been passed
if [ $# -lt 1 ]; then
  echo "Usage: ./clean.sh <dir>"
  exit 1
fi

# Find files
find $1 -type f \( -name ".*" -or -name "*~" \) > tmp.txt

# Erase files with an even length number, rename those with an odd length name
while read line; do
  name=$(basename $line)
  n=$(echo $name | wc -m)
  let "even=n%2"
  if [ $even -eq 0 ]; then
    rm $line
  else
    mb $line "$line.odd"
  fi
done < tmp.txt

# Clean temporary file
rm tmp.txt

exit 0


#!/usr/bin/env bash
################################################################################
# Version 3: Using find in pipe with egrep and ${#}
################################################################################

# Check if the correct parameters have been passed
if [ $# -lt 1 ]; then
  echo "Usage: ./clean.sh <dir>"
  exit 1
```

```
fi

# Find files
find $1 -type f | egrep -E ".*/(\..*)|(.*~)$" > tmp.txt

# Erase files with an even length number, rename those with an odd length name
while read line; do
  name=$(basename $line)
  n=${#name}
  let "even=n%2"
  if [ $even -eq 0 ]; then
    rm $line
  else
    mv $line $line".odd"
  fi
done < tmp.txt

# Clean temporary file
rm tmp.txt

exit 0


#!/usr/bin/env bash
################################################################################
# Version 4: Using two finds in pipe with egrep and xargs
################################################################################

# Check if the correct parameters have been passed
if [ $# -lt 1 ]; then
  echo "Usage: ./clean.sh <dir>"
  exit 1
fi

# Remove files that start with . and/or end with ~ and have a name of even lenght
find $1 -type f \( -name ".*" -or -name "*~" \) | egrep -E ".*/(..)+$" | xargs rm

# Rename files that start with . and/or end with ~ and have a name of odd lenght
find $1 -type f \( -name ".*" -or -name "*~" \) | egrep -E ".*/.(..)+$" | xargs -I
{} mv {} {}".odd"

exit 0


#!/usr/bin/env bash
################################################################################
# Version 5: Using recursive calls to the script to navigate the directory structure
################################################################################

# Check if the correct parameters have been passed
if [ $# -lt 1 ]; then
  echo "Usage: ./clean.sh <dir>"
  exit 1
fi

# Needed to scan hidden files and ignore . and .. entries
GLOBIGNORE=".:.."
```

```
# Scan entries in the current directory
for entry in $1/*; do
  if [ -d "$entry" ]; then
    $0 "$entry"
  else
     name=$(basename $entry)
     echo $name | egrep -e "^\..*" -e ".*~$" > /dev/null
     if [ $? -eq 0 ]; then
       n=${#name}
       let "even=n%2"
       if [ $even -eq 0 ]; then
         rm $entry
       else
         mv $entry $entry".odd"
       fi
     fi
  fi
done

exit 0
```

Ex 11 (3.5 points)

Italiano

Si indichino le principali differenze tra processi e thread. Si descriva la gestione di tali entità da parte del sistema operativo (struttura in memoria, ecc.). Se ne indichino le caratteristiche principali e i relativi vantaggi e svantaggi.

Si indichino inoltre le principali differenze tra thread a livello utente e thread a livello kernel, descrivendo i vari modelli di thread normalmente disponibili.

English

Indicate the main differences between processes and threads. Describe how these entities are handled by the operating system (data structures, etc.). State the main characteristics of both along with the advantages and disadvantages of using each of them. In addition, indicate the main differences between thread at user level and threads at kernel level describing the multi thread programming models usually available.

Risposta: Answer:

- Processo: Process:
  - creazione lenta slow creation
  - spazio di indirizzamento duplicato duplicate address space
  - gerarchia padre - figlio hierarchy parent - child
  - gestione attraverso PCB (Process Control Block) e diagramma stati dei processi. Managed through PCB (Process Control Block) and process states diagram.
- Thread: Thread:
  - creazione più veloce fast creation
  - spazio di indirizzamento condiviso shared address space
  - dati privati: program counter, registri, stack private data: program counter, registers, stack
  - maggiore scalabilità greater scalability
  - richiedono maggiori sforzi di sincronizzazione. require more synchronization efforts.
- Kernel thread: Kernel thread
  - thread gestiti dal kernel tramite system call threads managed by the kernel through system call
  - esiste una thread table in ogni processo (TCB simile alla PCB) exists a thread table for any process (TCP similar to PCB)
  - la schedulazione è gestita dal kernel scheduling is managed by the kernel
  - problemi su thread bloccanti problems on blocked threads
  - effettivo parallelismo. effective parallelism.

- User thread: User thread:
    - il kernel gestisce solo i processi; i thread sono gestiti da una libreria utente the kernel manage only processes; threads are managed by a user library and the kernel manages only processes;
    - le informazioni sui thread sono memorizzate all'interno dei processi information about threads are stored within processes
    - i kernel thread sono mappati su user thread. kernel threads are mapped on user threads.
- Implementazione ibrida: hybrid implementation
    - intermedia e comunemente adottata dai moderni SO. intermediate and commonly adopted by modern SOs.