Thread:

- A Thread is a small unit of process.
- It is a lightweight function that can run same time as other code.
- Thread allows a program to do multiple tasks at same time.

Why Need:

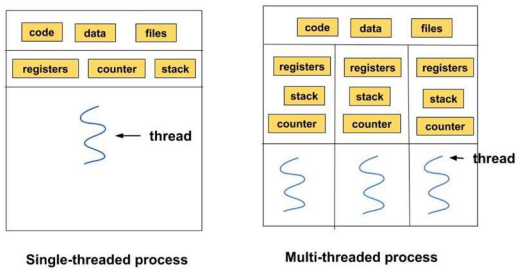
- To do multiple task together.
- To make the program faster and more.
- To use less memory.

How work in C?

- In c use POSIX Thread(pthread) libary.
- Create function for the Thread.
- Call pthread_create() to start the thread.
- Pthread_join() to wait for the thread to finish.

Multithreading:

Multithreading is a programming technique that allows multiple parts of a program, called threads, to execute concurrently within a single process.



SCALER Topics

- **In C programming language, we use the POSIX Threads (pthreads) library to implement**

Header File:

<pthread.h>

Creating a Thread

```
pthread_create(thread, attr, routine, arg);
```

thread	A place to store the ID of the new thread. Think of it like a name tag for the thread.
attr	Settings for the thread. Use NULL if you want the default settings.
routine	The function the thread will run. It must return void* and take void* as input.
arg	The input you want to give to the thread function. Use NULL if no input is needed. You can also pass a pointer to a struct to send multiple values.

Thread Function :

pthread_create ->start a new thread.

pthread_join ->wait for a thread to finish.

pthread_exit() ->end the thread.

pthread_t ->Data type for thread variable.

Basic Code :

```c
#include <stdio.h>
#include <pthread.h>
void *print_even(void *arg){
  pthread_t tid=pthread_self();
  for(int i=0;i<=10;i+=2){
    printf("Thread %lu - Even : %d\n",tid,i);
  }
  return NULL; //*Thread Terminated
}
void *print_odd(void *arg){
   pthread_t tid=pthread_self();
   for(int i=1;i<10;i+=2){
    printf("Thread %lu - Odd :%d\n",tid,i);
   }
   //return NULL;
   pthread_exit(NULL); //*Thread terminates here
}
int main() {
    //Code
    pthread_t t1,t2;
    pthread_create(&t1,NULL,print_even,NULL);
    //pthread_cancel(t1);
    pthread_create(&t2,NULL,print_odd,NULL);

    pthread_join(t1,NULL);
    pthread_join(t2,NULL);
    return 0;
}
```

📝 Purpose of the Code

This program creates **two threads**:

- One prints **even numbers** from 0 to 10.
- The other prints **odd numbers** from 1 to 9. Each thread prints its **own thread ID** along with the number.

🧠 Key Concepts Used

- **Multithreading**: Running two functions at the same time
- **Thread ID**: Identifies each thread
- **Thread termination**: Using return NULL or pthread_exit(NULL)
- **Synchronization**: pthread_join() ensures main waits for threads