**Chapter 9: MULTITHREADING**

**Topic – 1: Creation Of Thread**

**Introduction**

* There are two types of multitasking – **Process based** & **thread based**

**Steps Involved**

* **Step 1:** Create a function of void \* type.
* **Step 2:** Create **thread ID** storage in **main()** function.
* **Step 3:** Create the thread.
* **Step 4:** Add a **join function** which allows thread to finish execution.

**Code Example**

***/\* Step 1 \*/***

***void \*func(void \*arg) // At least one void\* type arg is compulsory***

***{***

***pthread\_exit(NULL); // NULL is exit status***

***return NULL;***

***}***

***int main()***

***{***

***/\* Step 2 \*/***

***pthread\_t thread; // Thread ID***

***pthread\_t thread[7]; // Array of thread IDs***

***/\* Step 3 \*/***

***pthread\_create(&thread, NULL, func, NULL);***

***// NULL – Thread attributes***

***// func – Function to call***

***// NULL – Argument of function***

***/\* Step 4 \*/***

***pthread\_join(thread, NULL);***

***// NULL – Return value to thread’s function***

**Topic – 2: Thread Attributes**

**Monologue**

* Attribute’s **lifecycle** is similar to **thread** itself, but both are different things.
* Making attribute argument in **pthread\_create()** as **NULL** means setting **default** configuration for our thread.

**Thread Attribute Life Cycle**

* Creation
* Initialization
* State definition
* Death

**Creation**

***pthread\_attr\_t attribute;***

**Initialization**

***pthread\_attr\_init(&attribute);***

**State Definition**

***// Joinable state***

***pthread\_attr\_setdetachstate(&attribute, PTHREAD\_CREATE\_JOINABLE);***

***// Detached state***

***pthread\_attr\_setdetachstate(&attribute, PTHREAD\_CREATE\_DETACHED);***

**Stack Size Setting**

***size\_t size = 2\*1024\*1024 // 2 MB stack size***

***pthread\_attr\_setstacksize(&attribute, size);***

**Death**

***pthread\_attr\_destroy(&attr);***

**Topic – 4: Mutex**

**Introduction**

* **Mutex:** Stands for ***mutually exclusive***.
* Used for preventive **multiple threads** from accessing **commonly shared** resources.

**Mutex Lifecycle**

* Creation
* Initialization
* Action
* Destroy

**Creation**

***pthread\_mutex\_t m;***

**Initialization**

***pthread\_mutex\_init(&m, NULL); // NULL – Mutex attribute***

**Action**

***// Blocks until critical section is locked by another thread***

***pthread\_mutex\_lock(&m);***

***// Unlocks the thread***

***pthread\_mutex\_unlock(&m);***

***// Returns 0 if mutex is already locked***

***pthread\_mutex\_trylock(&m);***

**Death**

***pthread\_mutex\_destroy(&m);***

**Topic – 5: Mutex Attribute**

**Mutex Attribute Lifecycle**

* Creation
* Initialization
* Set type

**Creation**

***pthread\_mutexattr\_t myAttr;***

**Initialization**

***pthread\_mutexattr\_init(&myAttr);***

**Set Type**

***// This one is default one, we will discuss more mutex attributes soon***

***pthread\_mutexattr\_settype(&myAttr, PTHREAD\_MUTEX\_NORMAL);***