Name: Tazmeen Afroz
Section: BAI-5A

Roll No: 22P-9255
Course: Operating Systems

Assignment # 3

Q#1

What resources are used when a thread is created? How do they differ from those used when a process is created?

Thread creation is a lighter operation compared to process creating due to smaller resource footprint.

Thread Creation:

- Involves allocating a small data structure that includes:
  - Register set
  - Stack
  - Priority

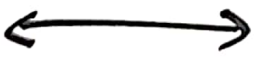
Process Creation:

- Allocating a large and more complex datastructure PCB (Process Control Block)

PCB includes:

- Memory Map
- Open files
- Environment variables.

The most time-consuming task in process creation is managing the memory map, which is not necessary for thread creation. As a result, thread creation is typically faster and more resource-efficient



Q#2 Using Amdahl's law, calculate the speedup ... 60 percent parallel component for (a) 2 processing cores (b) 4 processing cores (c) 8 processing cores.

$$P = 1 - S$$
$$S = 1 - P$$
$$\text{Speedup} = \frac{1}{S + \left(\frac{P}{N}\right)} = \frac{1}{(1-P) + \left(\frac{P}{N}\right)}$$

P → parallel portion → 60% → 0.60
N → number of cores

(a) 2 cores

$$= \frac{1}{(1-0.60) + \left(\frac{0.60}{2}\right)} = \frac{1}{0.4 + 0.3} = \frac{1}{0.7} = \boxed{1.43}$$

(b) 4 cores

$$= \frac{1}{(1-0.60) + \left(\frac{0.60}{4}\right)} = \frac{1}{0.4 + 0.15} = \frac{1}{0.55} = \boxed{1.82}$$

(c) 8 cores

$$= \frac{1}{(1-0.60) + \left(\frac{0.60}{8}\right)} = \frac{1}{0.4 + 0.075} = \frac{1}{0.475} = \boxed{2.1}$$

Question # 3
Which of the following components ..... memory.

Ans (b) Heap memory
(c) Global variables.

Heap memory: Shared by all threads.
Global variables. Accessible to all threads within the process
Each thread has its own set of registers and its own stack.

Question : 4

```
cd "/home/tazneen/OS_theory/" && g++ first.cpp -o first && "/home/tazneen/OS_theory/"first
(base) tazneen@tazneen:~/OS_theory$ cd "/home/tazneen/OS_theory/" && g++ first.cpp -o first && "/home/tazneen/OS_theory/"first
Factorial of 5 is 120
Sum of 3 and 7 is 10
Fibonacci series up to 10: 0 1 1 2 3 5 8 13 21 34
```

The program perform three calculations concurrently:

1. **Calculates the factorial** of 5, resulting in Factorial of 5 is 120.
2. **Calculates the sum** of 3 and 7, outputting Sum of 3 and 7 is 10.
3. **Generates the Fibonacci series** up to 10, printing 0 1 1 2 3 5 8 13 21 34.

Although the Fibonacci thread was created before the sum function, the execution does not depend on the order of creation, as the sum function executed first. This demonstrates the concurrent nature of multi-threading, where the faster computation can finish before the slower one, leading to nondeterministic output order.

Question : 5

```
(base) tazmeen@tazmeen:~/OS_theory$ cd "/home/tazmeen/OS_theory/" && g++ second.cp
Sum of 8 and 4 is 12
Product of 8 and 4 is 32
Difference of 8 and 4 is 4
(base) tazmeen@tazmeen:~/OS_theory$
```

This program perform three arithmetic operations concurrently using POSIX threads (pthreads):

1. **Addition** of 8 and 4, resulting in `Sum of 8 and 4 is 12`.
2. **Multiplication** of 8 and 4, giving `Product of 8 and 4 is 32`.
3. **Subtraction** of 8 and 4, yielding `Difference of 8 and 4 is 4`.

Although the subtraction thread was created before the product thread, the execution order does not depend on the order of creation, as the product or difference functions may finish first. This illustrates the concurrent nature of multi-threading, where operations are performed simultaneously, leading to potentially varying output sequences.