4)

سوال 2:

در این سوال از multi-process programming استفاده می کنیم.

به این صورت که با دستور ()fork رکویست های جدید را در process جدید اجرا میکنیم و چون دستور sleep(5) فقط در child process اجرا میشود کل رکویست ها باهم 5 ثانیه زمان می برند.

در این روش child از متغیر های parent استفاده میکند ولی هر تغییری روی متغیر بعد از دستور ()fork انجام بشود، روی متغیر های پردازنده های جدید اجرا نمی شود.

مزایا:

در این روش هر تعداد رکویست فرستاده شود، ریکوئست بعدی با سرعت انجام میشود و فقط یک رکوئست 5 ثانیه sleep می کند.

کد آن ساده و مفهوم است.

معایب:

برای جابجایی داده ها بین process ها، به memory و overhead بیشتری نسبت به thread ها نیاز دارد.

کل memory در هر subprocess کپی می شود، که می تواند overhead بیشتری ایجاد کند.

سوال 3:

در این سوال از multi-thread programming استفاده می کنیم.

به این صورت که هر دفعه رکوئست جدید داده می شود یک thread جدید می سازیم و تابع مورد نظر در این thread جدید اجرا می شود و ID هر thread در آرایه ذخیره می شود . در این روش همه رکوئست ها به صورت موازی زمان می برند و مستقل از هم هستند. به ازای هر کدام 5 ثانیه sleep اجرا می شود.

مزایا:

در اینجا thread ها فقط به متغیر های گلوبال و ورودی ها دسترسی دارند و overheadکمتری دارند.

این روش به memory کمتری نیاز دارد.

دسترسی به memory آسان‌تر است زیرا thread ها process یکسانی را به اشتراک می‌گذارند.

جابجایی بین thread ها سریع است.

تولید thread جدید در یک process موجود سریعتر از ایجاد یک process کاملاً جدید است.

همه thread ها یک مخزن حافظه پردازشی و فضای آدرس یکسانی دارند.

معایب:

سرعت کم زیرا هر رکویست 5 ثانیه طول میکشد .

کد آن کمی چالشی تر است.

overhead مربوط به مدیریت thread های مختلف برای کارهای basic پرهزینه است.