1. Parâmetros
    a. getcontext(ucontext_t *ucp) - função responsável por inicializar a variável a do tipo ucontext_t, utilizada para armazenar informações e referências a contextos, a partir do contexto ativo naquele momento
    b. setcontext(ucontext_t *ucp) - a função recebe uma variável do tipo ucontext_t e tem por objetivo restaurar o contexto referenciado através desta variável, que foi inicializada através de uma chamada de getcontext() ou makecontext.
    c. swapcontext(ucontext_t *ucp, ucontext_t *ucp) - salva o contexto atual na primeira variável a do tipo ucontext_t e ativa o contexto referenciado na segunda variável b do tipo ucontext_t
    d. makecontext(ucontext_t *ucp, void (*func)(), int argc, ...) - Altera o contexto apontado pela variável a, obtido através de uma chamada da função getcontext(). Para que isto ocorra uma pilha deve ser alocada e assinalada para o atributo uc stack e um contexto sucessor deve ser assinalado para uc_link. Uma vez ativado este contexto a função func é chamada, passando seus n argumentos contados através de argc. Uma vez retornada a função o contexto sucessor é ativado, e caso seja nulo a thread é encerrada
2.
    a. uc_stack.ss_sp - Aponta para a base da região de memória alocada para a pilha
    b. uc_stack.ss_size - Indica o tamanho da região alocada para a pilha
    c. uc_stack.ss_flags = 0;
    d. uc_link - Refere-se ao contexto sucessor que sera ativado uma vez que a função referente ao contexto atual retornar
3.
    a. getcontext (&ContextPing); - Ping é incializado com contexto atual
    b. ContextPing.uc_stack.ss_sp = stack ;
    ContextPing.uc_stack.ss_size = STACKSIZE;
    ContextPing.uc_stack.ss_flags = 0;
    ContextPing.uc_link = 0;
    - As variavies relativas a pilha de ping são incializadas
    c. makecontext (&ContextPing, (void*)(*BodyPing), 1, " Ping"); - O contexto de ping é alterado e a função BodyPing é designada para ser executada uma vez que ping for ativado, recebendo 1 parametro = " Ping"
    d. getcontext (&ContextPing); - Ping é incializado com contexto atual
    e. ContextPong.uc_stack.ss_sp = stack ;
    ContextPong.uc_stack.ss_size = STACKSIZE;
    ContextPong.uc_stack.ss_flags = 0;
    ContextPong.uc_link = 0;
    - As variavies relativas a pilha de ping são incializadas
    f. makecontext (&ContextPong, (void*)(*BodyPong), 1, " Pong"); - O contexto de ping é alterado e a função BodyPong é designada para ser executada uma vez que ping for ativado, recebendo 1 parametro = " Pong"

g. swapcontext (&ContextMain, &ContextPing); - O contexto atual é salvo na variável ContextMain e o contexto de ContextPing é ativado
h. swapcontext (&ContextMain, &ContextPong); O contexto atual é salvo na variável ContextMain e o contexto de ContextPong é ativado
i. swapcontext (&ContextPing, &ContextPong); - O contexto atual é salvo na variável ContextPing e o contexto de ContextPong é ativado
j. swapcontext (&ContextPong, &ContextPing); - O contexto atual é salvo na variável ContextPong e o contexto de ContextPing é ativado
k. swapcontext (&ContextPing, &ContextMain) ; - O contexto atual é salvo na variável ContextPing e o contexto de ContextMain é ativado
l. swapcontext (&ContextPong, &ContextMain) ; - O contexto atual é salvo na variável ContextPong e o contexto de ContextMain é ativado