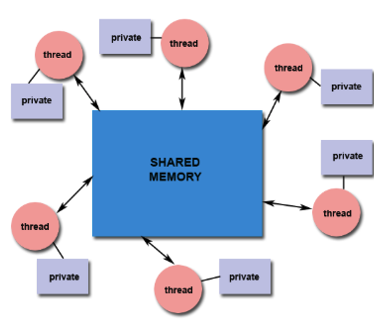
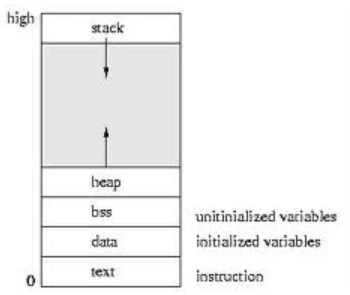
1. Napišite tip podataka prvog parametra kod funkcije koja je ulazna točka u niti
   1. Pvoid
2. Povežite funkcije za upravljanje s memorijskim heap-ovima s njihovom namjenom
   1. HeapFree 🡪 Oslobađa memoriju zauzetu s HeapAlloc
   2. HeapAlloc 🡪 Zauzima memoriju iz navedenog heap-a
   3. HeapWalk 🡪 Prebrojava sve blokove zauzete memorije unutar heap-a
   4. HeapValidate 🡪 Za odre]ivanje da li je heap ispravan/oštećen
   5. HeapDestroy 🡪 Uništava korisnički heap kojeg smo sami napravili s CreateHeap
3. Skraćenica TLS kod procesa i niti znači:
   1. Thread local storage
4. Poredajte klase prioriteta procesa od najniže do najviše (Manji broj znači niži prioriteti).
   1. IDLE\_PRIORITY\_CLASS 🡪 4
   2. NORMAL\_PRIORITY\_CLASS 🡪 8
   3. NORMAL\_PRIORITY\_CLASS + za niti THREAD\_PRIORITY\_HIGHEST 🡪 10
   4. HIGH\_PRIORITY\_CLASS 🡪 13
   5. REALTIME\_PRIORITY\_CLASS 🡪 24
5. Koliko imamo osnovnih klasa prioriteta procesa kod Windowsa?
   1. 4
6. S obzirom na sliku, označite tvrdnje koje su točne.
   1. Dio memorije 'private' je zapravo TLS.
   2. Dio memorije 'shared memory' pripada procesu, a 'private' pripada svakoj od niti.



1. Označite tvrdnje koje su točne, vezane za API funkciju OpenThread:
   1. Ta funkcija ima 3 parametra.
   2. Pomoću ID-a od niti možemo dobiti njen HANDLE.
   3. Ta funkcija vraća vrijednost tipa HANDLE.
   4. Ova funkcija spada u skupinu Windows API funkcija za upravljanje nitima.
2. Dana je struktura STARTUPINGO koja se koristi prilikom kreiranja procesa:
   1. Napisati koliko byte-ova zauzima u memoriji ova struktura podataka, ako se zna da su kod Win 32 platforme svi pokazivači i handlovi 32-bitni. WORD je 16-bitni podatak. 🡪 68



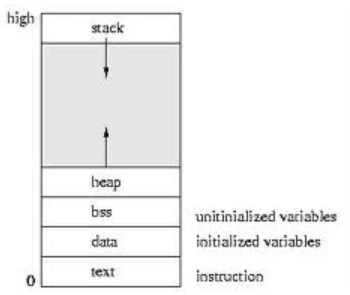
1. Kako se zove koncept u kojem Windows koristi tvrdi disk kao dodatno memorijsko spremište za proces i podatke.
   1. virtualna memorija
2. Koliko byte-ova je velika memorijska adresa kod Win32?
   1. 4
3. Napišite ime Windows API funkcije s kojom se može startati Windows proces.
   1. CreateProcess
4. Kojom kraticom označavamo datoteke koje možemo dinamički povezati s našom aplikacijom kao dodatne biblioteke s kodom i podacima.
   1. dll
5. Čekamo da više niti završi pozivajući funkciju WaitForMultipleObjects. Ova funkcija može vratiti različite vrijednosti. Povežite moguću vraćenu vrijednost sa scenarijem koji se dogodio.
   1. WAIT\_ABANDONED 🡪 Handle-ovi nisu bili ispravni pa nije moguće čekati s ovom funkcijom.
   2. WAIT\_FAILED 🡪 Čekanje je završilo na neubičajeni način. Treba dodatno povezati GetLastError za kod greške.
   3. WAIT\_TIMEOUT 🡪 Funkcija je završila jer je vrijeme za čekanje isteklo.
   4. WAIT\_OBJECT\_0+2 🡪 Čekanje je završilo tako da je 3. niti u redu čekanja jedina završila.
   5. WAIT\_ OBJECT\_0 🡪 Čekanje je završilo tako da su sve niti završile.
6. Ukoliko ne navedemo veličinu stoga prilikom kreiranja niti funkcijom CreateThread, koliko se onda preddefinirano uzima za veličinu stoga?
   1. 1MB
7. Povežite navedene izraze i funkcije s načinom korištenja DLL-a.
   1. \_declspec (dllimport) 🡪 kod implicitnog povezivanja DLL-a
   2. \_declspec (dllexport) 🡪 kod implicitnog povezivanja DLL-a
   3. LoadLibrary 🡪 kod explicitnog povezivanja DLL-a
   4. FreeLibrary 🡪 kod explicitnog povezivanja DLL-a
   5. GetProcAddress 🡪 kod explicitnog povezivanja DLL-a
8. Ako ime izvršne slike prilikom startanja procesa ne sadrži punu putanju onda se pretražuju neke posebne lokacije gdje bi se program mogao nalaziti. Pridijelite točan redoslijed pretraživanja izvršne slike programa u tom slučaju:
   1. Direktorij izvršne slike tekućeg procesa 🡪1
   2. Tekući direktorij 🡪 2
   3. Winows sistemski direktorij 🡪 3
   4. Windows direktorij 🡪 4
   5. Svi direktoriji određeni varijablom okruženja PATH 🡪 5
9. Koji tip podataka vraća funkcija za startanje procesa, za razliku od većine funkcija koje kreiraju osnovne objekte jezgre kod Windows-a?
   1. Bool
10. Windows API funkcija s kojom možemo postaviti prioritet niti jeste:
    1. SetPriorityClass
11. Treba usporediti stanja procesa/niti u Linux i Windows okruženju i povezati odgovarajuća stanja u skaldu sa slikom.
    1. Windows: Running 🡪 Linux: Executing
    2. Windows: Ready 🡪 Linux: Ready
    3. Windows: Terminated 🡪 Linux: Zombie
    4. Windows: Wiaiting 🡪 Linux: Stopped/Interruptible/Uninterruptible
12. Koji tip podataka vraca funkcija za kreiranje Niti, kao i vecina drugi Windows API Create ... funckija?
    1. HANDLE
13. Najbrže zautimanje memorije se dobiva ako se postave zastavice HEAP\_NO\_SERIALIZE i HEAP\_GENERATE\_EXCEPTIONS na sljedeći način:
    1. Postavimo samo HEAP\_NO\_SERIALIZE
14. Odrediti s kojim znakom se razdvajaju parametri komandne linije kada startamo proces s parametrima (argumentima).
    1. “ ” – razmak
15. Napišite ime funkcije s kojom drugi proces može dobiti HANDLE memorijski mapirane datoteke koji je stvorio prvi proces s funkcijom CreateFileMapping. (Drugi proces bi u ovom slučaju morao znati ime pod kojim je prvo proces kreirao objekt memorijski mapirane datoteke.)
    1. OpenFileMapping
16. Što sve mora imati Windows proces?
    1. Procesni heap
    2. Jednu nit
    3. Stringove okruženja
17. Vrijednost „high“ na slici koja prikazuje memorijsku arhitekturu procesa kod Win32 može biti najviše (GB):
    1. 2



1. Prilikom deklaracije funkcije koja je ulazna točka u niti između tipa podataka koja ona vraća DWORD i imena te funkcije moramo napisati poseban specifikator deklaracije.
   1. WINAPI
2. Koliko ima parametara funkcija za kreiranje niti CreateThread ?
   1. 6
3. Prilikom startanja procesa kao zadnji parametar možemo predati preddefiniranu strukturu PROCESS\_INFORMATION koja će se popuniti s potrebnim podacima. Koliko tih podataka ima ?
   1. 4
4. Mogu li i kako procesi dijeliti memoriju?
   1. Procesi mogu dijeliti memoriju samo na jedan način - korištenjem memorijski mapirane datoteke
5. Koliko parametara ima fukcija koja je ulazna točka u nit?
   1. 1
6. NIt iz stanja " izvodi se " ( running) prelazi u stanje  " spremna " kada se desi događaj kojeg na eng . nazivamo  "preempted".
   1. Vremenski odsječak izvođenja je gotov - nit se vraća u red čekanja.
7. Dan je slijedeći sistemski program. ???
   1. WaitForSingleObject ?

#include <windows.h>   
#include <stdio.h>   
#include <tchar.h>   
void \_tmain( int argc, TCHAR \*argv[] )  
{  
STARTUPINFO si;  
PROCESS\_INFORMATION pi;  
ZeroMemory( &si, sizeof(si) );  
si.cb = sizeof(si);  
ZeroMemory( &pi, sizeof(pi) );  
if( argc != 2 )  
{  
printf("Usage: %s [cmdline]\n", argv[0]);  
return;  
}  
// Start the child process.   
if( !CreateProcess( NULL, // No module name (use command line)  
argv[1], // Command line  
NULL, // Process handle not inheritable  
NULL, // Thread handle not inheritable  
FALSE, // Set handle inheritance to FALSE  
0, // No creation flags  
NULL, // Use parent's environment block  
NULL, // Use parent's starting directory   
&si, // Pointer to STARTUPINFO structure  
&pi ) // Pointer to PROCESS\_INFORMATION structure  
)   
{  
printf( "CreateProcess failed (%d).\n", GetLastError() );  
return;  
}  
WaitForSingleObject( pi.hProcess, INFINITE );  
CloseHandle( pi.hProcess );  
CloseHandle( pi.hThread );  
}

1. Povežite navedene funkcije za upravljanje nitima s njihovim značenjem.
   1. ExitThread 🡪 Završava pozivnu nit. Nit je u stanju *završena* nakon što je završena s ExitThread ili TerminateThread ali još uvijek može imati otvorenih handle-ova
   2. SuspendThread 🡪 Obustavlja navedenu nit.
   3. CreateThread 🡪 vraća kao rezultat i ID od niti i njen handle, Stvara nit za izvršavanje unutar virtualnog adresnog prostora pozivajućeg procesa.
   4. PauseThread 🡪 Pauzira izvođenje navedene niti, prethodno kreirane pomoću CreateThread(). Nit se može nastaviti pomoću ResumeThread().
   5. ResumeThread 🡪 Nit može uvećavati ili umanjivati brojač suspenzije od druge niti (Ako je **CREATE\_SUSPENDED**, nova nit će biti u stanju suspenzije, te će se moći nastaviti izvršavati tek nakon poziva **ResumeThread** funkcije)
2. Upravljanje procesom se vrši pomoću HANDLE-a procesa ili pomoću njegovog PID (process ID). Povežite pomoću čega se upravlja navedenim funkcijama :
   1. ExitProces 🡪 pomoću HANDLE-a
   2. WaitForSingleObject 🡪 pomoću HANDLE-a
   3. TerminateProcess 🡪 Pomoću HANDLE-a
   4. OpenProcess 🡪 pomoću PID
   5. GetExitCodeProcess 🡪 pomoću HANDLE-a
3. Kako se zove osnovna jedinica izvršavanja pod Windows , osnovnija od procesa. Ona sadrži priatni prostor za spremanje podataka , ima svoj stog.
   1. Nit
4. Kako se zove Windows API funkcija za zaustavljanje drugih procesa iz nekog procesa?
   1. Terminate process
5. Povežite načine povezivanja biblioteka funkcija s njihovim opisom.
   1. DLL- implicitno povezivanj - Biblioteke funkcija se povezuju prilikom učitavanja izvršene slike programa
   2. DLL - eksplicitno povezivanje - Biblioteke funkcija se povezuju prilikom izvođenja programa
   3. Statičko povezivanje - Sve biblioteke funkciaj se povezuju u zajedničku izvršnu sliku(.exe)
6. Napišite ime funkcije iz C-bibliteke koja obavlja posao koji radi windows API HeapAlloc
   1. Malloc
7. S obzirom na navedenu sliku označite tvrdnje koje su točne.
   1. "text" dio na slici predstavlja dio memorije u kojem se nalazi kod programa i potprograma
   2. Kao i stack, i heap može biti rastući



1. Povežite funkciju s API skupinom kojoj pripada.
   1. GetProcessHeap 🡪 Heap API
   2. UnmapViewOfFile 🡪 MMF API
   3. HeapCreate 🡪 Heap API
   4. CreateFileMapping 🡪 MMF API
   5. CreateViewOfFile 🡪MMF API
2. A=broj parametara Windows API funkcije za startanje procesa. B= broj parametara C-bibliotečne funkcije za startanje procesa (system(..)). Koliko je A-B?
   1. 9
3. Označite točne tvrdnje vezane za memorijske heap-ove:
   1. Samo podaci za koje dinamički zauzimamo memoriju nalaze se u heap-u
   2. Svaki proces dobiva svoj " defaut" heap
   3. Podaci koje proces koristi nalaze se na memorijskom stogu i u memorijskom heap-u.
4. Napišite tip podataka provg parametra kod funkcije koja je ulazna točka u nit
   1. \_declspec