Name: Klodjan Hidri
Am: 2726
login: klodjan.hidri@gmail.com

Tropos ektelesis ths askisis:
//

```
cd  test/               #  benoume sto fakelo test  apo to fakelo assign1_HY527
./compile.sh            #  compile ola ta arxeia .
```

istera mporoume na ta treksoume ena ena ta tests :

```
./sieve
./sieve2
./sieve4
./sink
./sort
./spin
./spin2
./spin3
```

telos kanoume clean

```
./clean.sh
```

//

Για την υλοποιηση της ασκησης χρησιμοποιω μια ουρα readyqueue για να κρατησω ολα τα threads ,ενα hashtable για ολα τα threads που κανουν join,τα systems calls timer_create() [για να μετρησω ολο το χρονο εκτελεσης] και signal() [για να κανω context switch οταν θα σηκωθει το σημα SIGALRM].

Για την λειτουργια των threads πρωτα κανουμε init οπου δεσμευουμε χωρο για το struct thread ,αρχικοποιουμε τις 2 δομες την ουρα και το hashtable,δημιουργουμε το χρονο και την συχνοτητα που θα σηκωθει το signal SIGALRM για να κανουμε context_switch και την συχοτητα αυτη την οριζω =1 millisecond οπου ανα 1 μsec θα εχουμε αλλαγει του thread οπου το τρεχον thread θα αφησει την CPU για το επομενο thread.

Στη συνεχεια υλοποιουμε την δημιουργεια του thread οπου αρχικοποιουμε καταλληλα το την στοιβα δεσμευοντας 32 KB χωρο. Η μορφη της στοιβας θα ειναι περιπου οπως φαινεται παρακατω : στην κορυφη της στοιβας τοποθετουμε την διευθυνση την __thrstart γιατι το _swtch.s θα καλεσει την __thrstart μεσω την εντολης ret οπου η λειτουργεια του ειναι να κανει pop() τα περιεχομενα της κορυφης της στοιβας για να τα γραψει στο register EIP (instruction pointer register) οπου κραταει δεικτει για τη εκτελεση της επομενη εντολης οπου στην δικια μας περιπτωση η επομενη εντολη που θα εκτελεστει θα ειναι η __thrstart.Αφηνουμε θεσης κενες η 8 bytes και αμμεσως μετα τοποθετουμε τις διευθυνσεις των args και func αντοιστοιχα.

Στοιβα: x86 32KB

	Mem addr	esp
_thrstart	32767	esp+0
-	32764	esp+4
-	32760	esp+8
args	32766	esp+12
func	32762	esp+16
...		
...		
...		
...		
…..	0	esp+32767

Μετα απο την αρχικοποιηση της στοιβας του αντοιστιχου thread τοτε τοποθετειται στην readyqueue.Ολα τα threads θα τρεχουν το καθενα 1 millisecond και θα μπουν στην ουρα για να περιμενουν να ξαναεκτελεστουν.Οταν ενα thread καλει join τοτε αυτο το thread θα το βαλω στο hashtable με key το thread_id που καλεσε το join πχ αν Thread_A καλει join(0) το thread A θα μπει στο hashtable με key=0 (join_insert(0,Thread_A)) ,θα περιμενει ολα τα threads na τελειωσουν και στο τελος θα ξαναμπει στην ουρα για να εκτελεστει.Εαν το Thread_A καλει joint(thread_id !=0) τοτε το thread_A θα μπει στο hashTable θα περιμενει να τελειωσει μονο το thread_id που καλεσε στο join και θα ξαναμπει στην ουρα.

Οσον αφορα οταν ενα thread τελειωνει την εργασια του ελεγχω το hashtable με το id του αν εχει αλλο thread που τον περιμενει να τελειωσει αν εχει τοτε το φορτωνω στην ουρα για να εκτελεστει. Επισης για την περιπτωση που εχει γινει join(0) ελεγχω παντα αν η ουρα ειναι αδεια ωστε να το επαναφερω στην ουρα.