Πανεπιστήμιο Πειραιώς

Τμήμα πληροφορικης

Λειτουργικά Συστήματα

Εργασία 1η

Θέμα

Διεργασίες και νήματα

Φοιτητής

Όνομα: Αργυρίου Κωνσταντίνος
ΑΜ: Π19017
Email: argyrioukost@gmail.com

Αναλυτική εξήγηση του κώδικα

Το πρόγραμμα της εργασίας ξεκινάει αρχικά απο την συνάρτηση **main()** στην οποία καλούμε την κλήση συστήματος **fork()**, η οποία δημιουργεί μια θυγατρική διεργασία η οποία έχει ένα αντίγραφο του χώρου διευθύνσεων της διεργασίας πατέρα. Μετά από την επιτυχή κλήση της **fork** κάνουμε έναν έλεγχο για το εάν η διεργασία που εκτελείται αυτή την στιγμή είναι η διεργασία παιδί. Αν η διεργασία που εκτελείται είναι η διεργασία παιδί, τότε εκτελούμε την συνάρτηση **child_process_exec()**, αλλιώς αν είναι η διεργασία πατέρα καλούμε την κλήση συστήματος **wait()** για να περιμένουμε την ολοκλήρωση της εκτέλεσης της διεργασίας παιδί.

```
pid_t proc_id = fork();

if (proc_id == 0)
    child_process_exec();
else
    wait(0);
```

Η συνάρτηση **child_process_exec()** είναι η συνάρτηση που βγάζει εις πέρας τις λειτουργίες που πρέπει να εκτελέσει η διεργασία παιδί. Στην αρχή της συνάρτησης ορίζουμε τις μεταβλητές των νημάτων που θα χρησιμοποιήσουμε και τις μεταβλητές των χαρακτηριστικών αυτόν των νημάτων. Ύστερα χρησιμοποιείται η δομή **thread_run_info**, η οποία περιέχει πληροφορίες για το πως τρέχει το κάθε νήμα. Μετά αφού ορίστηκαν οι πληροφορίες που θα πρέπει να έχει κάθε νήμα, εκτελείται η κλήση συστήματος **pthread_create()**, με την οποία δημιουργούμε τα δύο ζητούμενα νήματα.

```
struct thread_run_info minus_t_run_info;
memcpy(&minus_t_run_info.time_to_run, &minus_t_rand, sizeof(int));
minus_t_run_info.what_to_exec = 0;
minus_t_run_info.is_exec_random = TRUE;
strcpy(minus_t_run_info.name, "minus");

struct thread_run_info increase_t_run_info;
memcpy(&increase_t_run_info.time_to_run, &increase_t_rand, sizeof(int));
increase_t_run_info.what_to_exec = 5;
increase_t_run_info.is_exec_random = FALSE;
strcpy(increase_t_run_info.name, "increase");

if (pthread_create(&minus_t, &minus_attr_t, &thread_run,
                   (void *)&minus_t_run_info) != 0)
FAILED_TO_CREATE_THREAD("minus")
if (pthread_create(&increase_t, &increase_attr_t, &thread_run,
```

```
                (void *)&increase_t_run_info) != 0)
FAILED_TO_CREATE_THREAD("increase")
```

Η συνάρτηση **thread_run()** είναι η συνάρτηση που είναι υπεύθυνη για την εκτέλεση και των δύο νημάτων, αυτού που μειώνει την number και αυτού που αυξάνει την number. Η συνάρτηση “καταλαβαίνει” ποια λειτουργία να εκτελέσει, πρόσθεση η αφαίρεση, με την χρήση των πληροφοριών που υπάρχουν στην δομή **thread_run_info**. Στην συνάρτηση **thread_run()** αρχικά λαμβάνεται ο, περίπου, χρόνος σε κύκλους ρολογιού που έχει δώσει ο επεξεργαστής στο πρόγραμμα μέχρι στιγμής και ύστερα σε έναν βρόχο αυτοί οι κύκλοι μετατρέπονται σε δευτερόλεπτα διαρόντας τους κύκλους με την σταθερά, και macro, της C **CLOCKS_PER_SEC** και κάθε νήμα εκτελείται για όσα δευτερόλεπτα περιγράφονται στην δομή **thread_run_info**, που έχουν παραχθεί τυχαία στην συνάρτηση **child_process_exec**. Επίσης για όσο τρέχει κάθε νήμα, εκτελείται η αντίστοιχη λειτουργία που περιγράφεται εξίσου στην δομή **thread_run_info**.

```
 do
{
      time_spend = clock() - thread_start_time;
      if (time_spend < 0) pthread_exit(0);

      if (thread_run_info->is_exec_random)
            change_number(-(1 + rand() % (10 + 1 - 1)));
      else
            change_number(thread_run_info->what_to_exec);

            curr_run_time = time_spend / CLOCKS_PER_SEC;
} while (curr_run_time < thread_run_info->time_to_run);
```

Τέλος στο τέλος της εκτέλεσης κάθε νήματος εκτελείται η συνάρτηση **print_thread_attr()** η οποία εμφανίζει τα χαρακτηριστικά κάθε νήματος και επίσης εμφανίζει και την αλλαγμένη απο το νήμα τιμή της μεταβλητής number.