

```

449 Request *select_request(Request *scheduled[], int N, int policy) {
450
451     Request *r = NULL;
452     if (N == 0) {
453         // no scheduled requests
454         return r;
455     }
456
457     switch (policy) {
458         case FCFS:
459             // returns the first scheduled request
460             r = scheduled[0];
461             break;
462         case SJF:
463             // returns the request with the smallest estimated time
464             r = scheduled[0];
465             int min_est_time = get_est_time(r);
466
467             for (int i = 0; i < N; i++) {
468                 if (get_est_time(scheduled[i]) < min_est_time) {
469                     r = scheduled[i];
470                     min_est_time = get_est_time(r);
471                 }
472             }
473             break;
474         case PES:
475             // returns the request with the highest priority
476             r = scheduled[0];
477             int max_priority = get_est_time(r);
478
479             for (int i = 0; i < N; i++) {
480                 if (get_est_time(scheduled[i]) > max_priority) {
481                     r = scheduled[i];
482                     max_priority = get_est_time(r);
483                 }
484             }
485             break;
486         default:
487             printf("Unknown scheduling policy.\n");
488             break;
489     }
490     return clone_request(r);
491 }

```