CS241 LAB

Midterm lab review

- Worksheet (please look at questions on the sheet
- quizizz
- Code review
- Coding Trick & Style workshop
- Fill up midterm review form

- Worksheet
- quizizz
- Code review
- Coding Trick & Style workshop
- Fill up midterm review form

P1 Program 1

```
int counter;

static void * foo(void * _tn){
  int i;
  for (i = 0; i < 100000; i++)
    counter++;
  return NULL;
}

int main() {
  int i, N = 5;
  pthread_t t[N];

  for (i = 0; i < N; i++)
    pthread_create(&t[i],
NULL, foo, NULL);

  for (i = 0; i < N; i++)
    pthread_join(t[i], NULL);

  printf("%d\n", counter);
  return 0;
}
```

Wrong

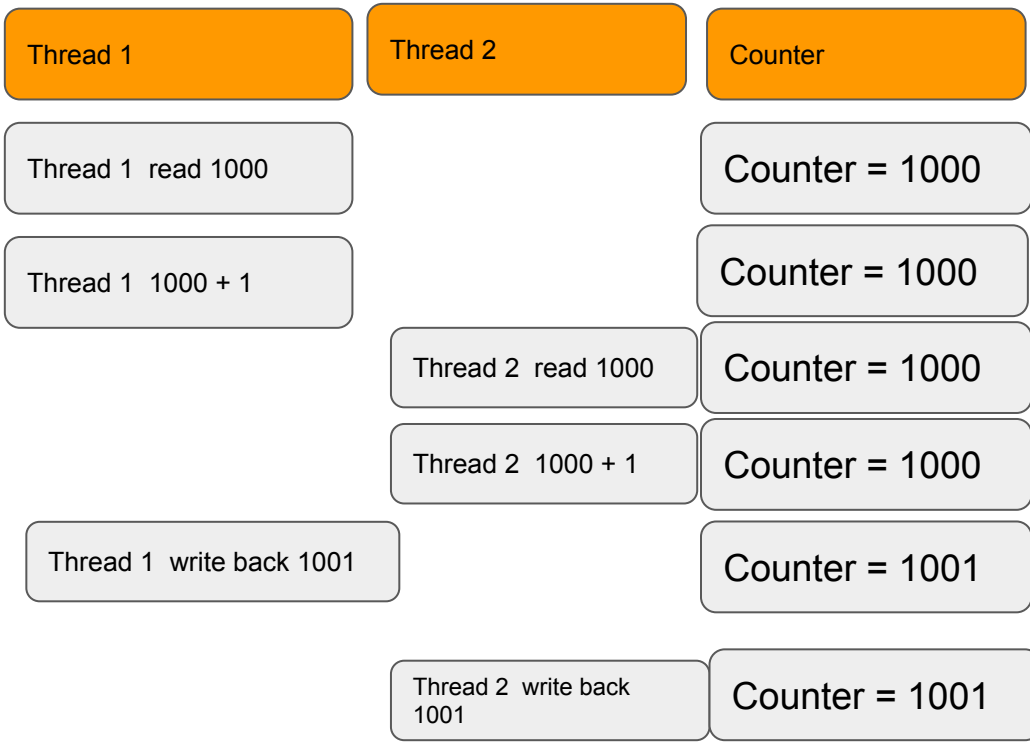
Good day!

Thread 1	Thread 2	Counter
Thread 1 read 1000		Counter = 1000
Thread 1 1000 + 1		Counter = 1000
Thread 1 write back 1001		Counter = 1001
	Thread 2 read 1001	Counter = 1001
	Thread 2 1001 + 1	Counter = 1001
	Thread 2 write back 1002	Counter = 1002

P1 Program 1

```
int counter;

static void * foo(void * _tn){
  int i;
  for (i = 0; i < 100000; i++)
    counter++;
  return NULL;
}

int main() {
  int i, N = 5;
  pthread_t t[N];

  for (i = 0; i < N; i++)
    pthread_create(&t[i],
NULL, foo, NULL);

  for (i = 0; i < N; i++)
    pthread_join(t[i], NULL);

  printf("%d\n", counter);
  return 0;
}
```

Bad day!



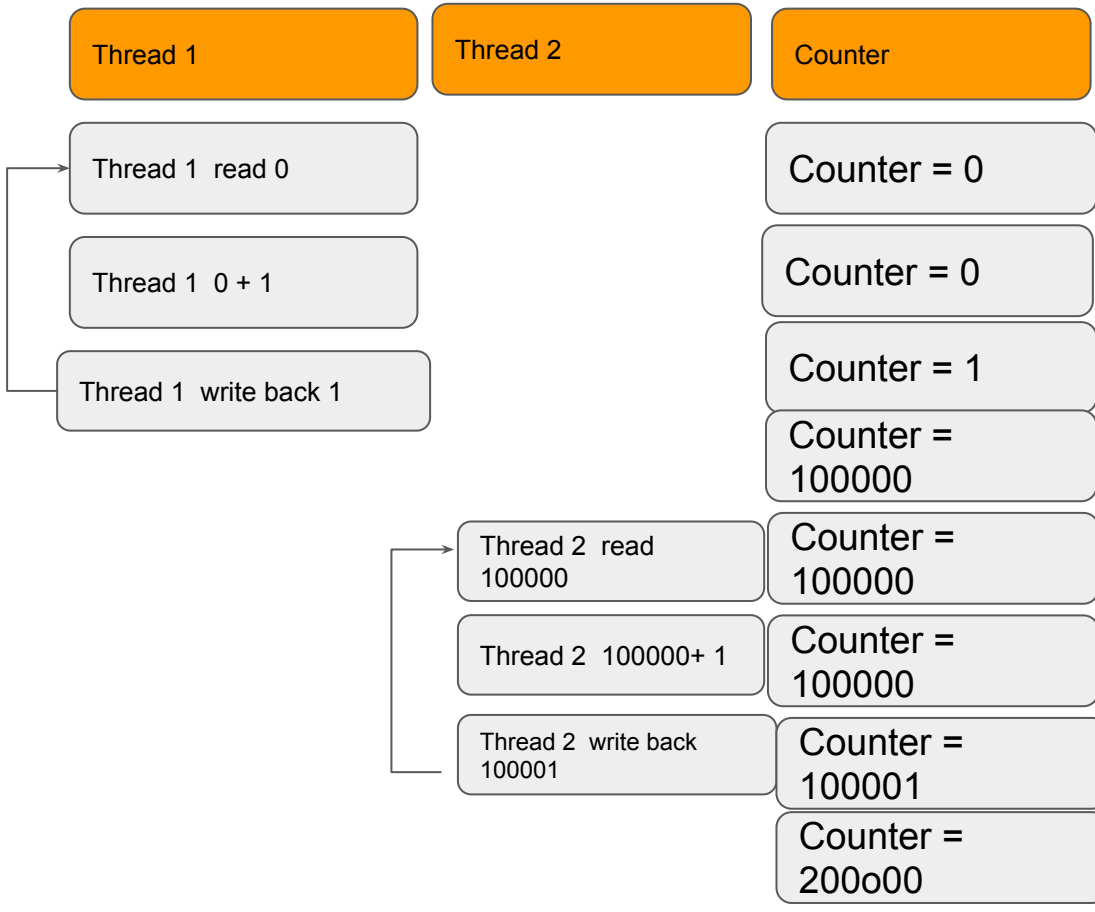
P1

Program 2

Right

```
int counter;

static void * foo(void * _tn) {
  int i;
  for (i = 0; i < 100000; i++)
    counter++;
  return NULL;
}

int main() {
  int i, N = 5;
  pthread_t t[N];

  for (i = 0; i < N; i++) {
    pthread_create(&t[i], NULL,
foo, NULL);
    pthread_join(t[i], NULL);
  }

  printf("%d\n", counter);
  return 0;
}
```

Everyday is a good day!

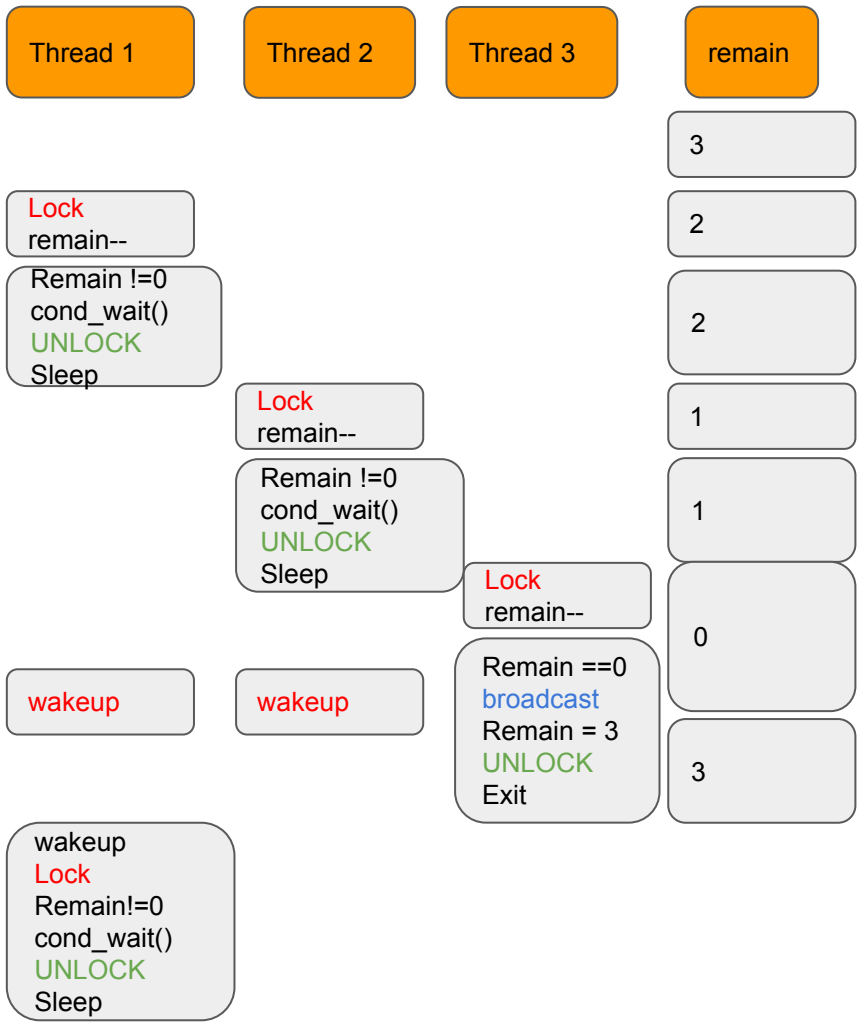


P2

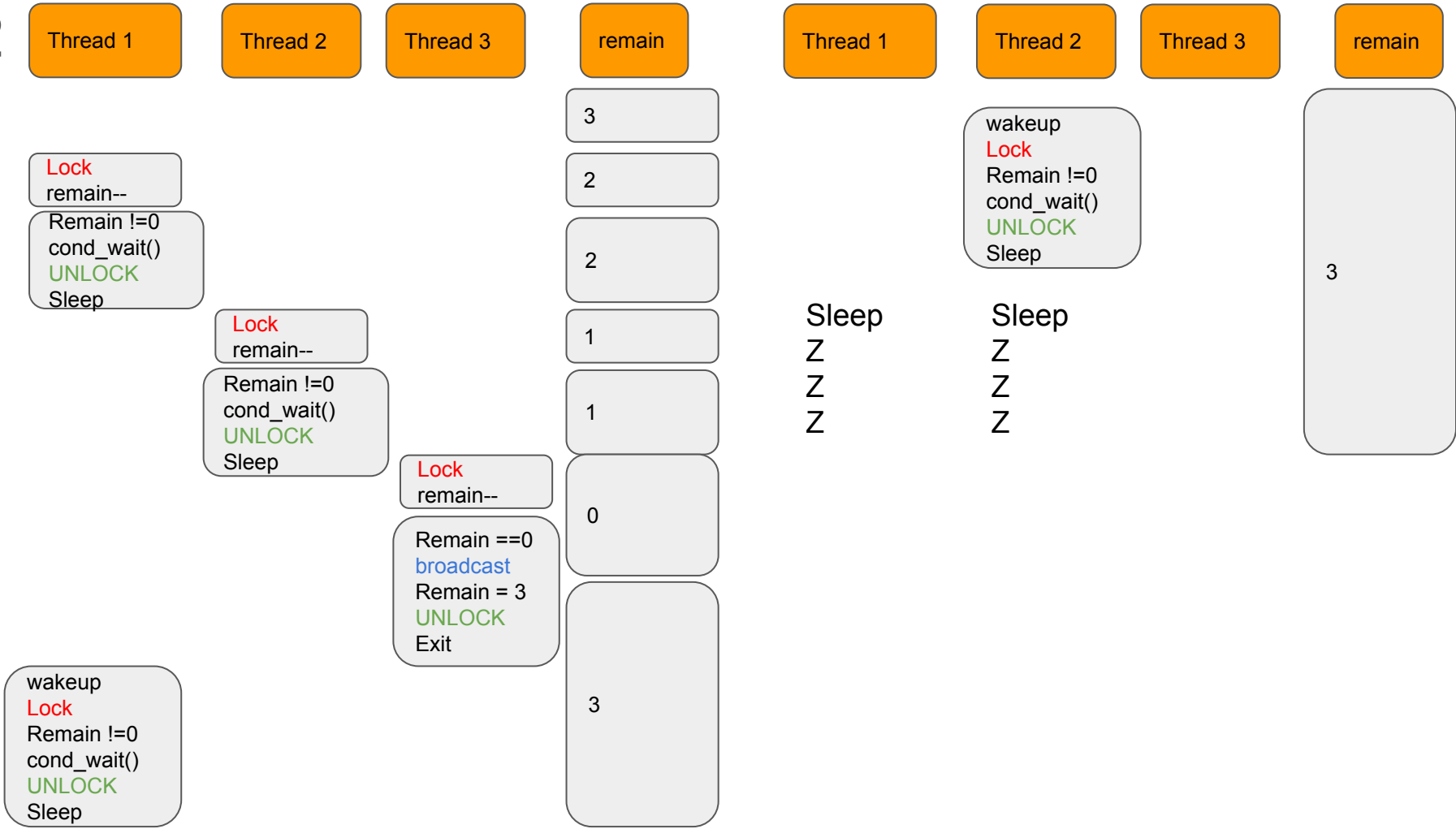
Wrong

```
pthread_mutex_lock(&m); remain--;
if (remain == 0) {
   pthread_cond_broadcast(&cv);
   remain = num_threads;
}
else {
   while(remain != 0) {
     pthread_cond_wait(&cv, &m);
   }
}
pthread_mutex_unlock(&m);
```

Unlock
Sleep
Lock



P2

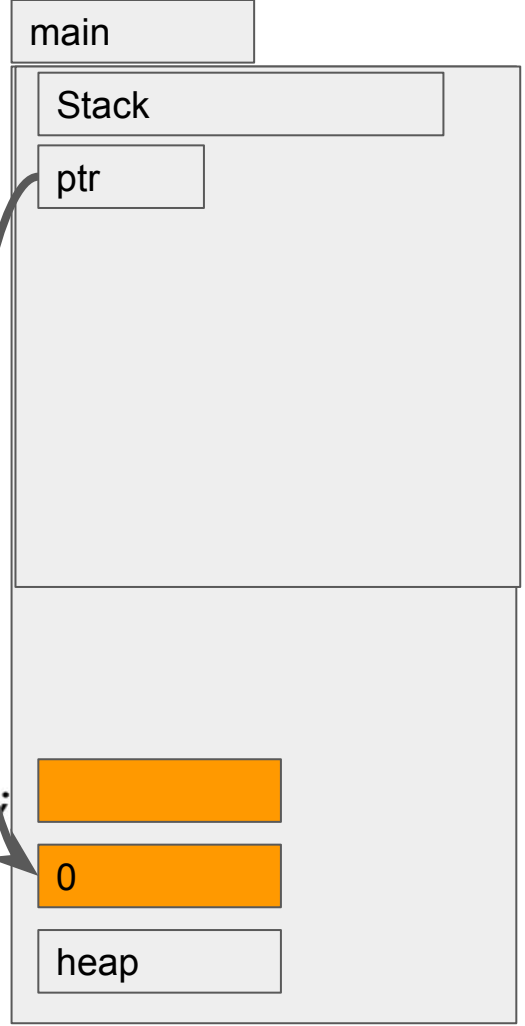

Thread 1
Thread 2
Thread 3
remain
Lock
remain--
Remain !=0
cond_wait()
UNLOCK
Sleep
Lock
remain--
Remain !=0
cond_wait()
UNLOCK
Sleep
Lock
remain--
Remain ==0
broadcast
Remain = 3
UNLOCK
Exit
wakeup
Lock
Remain !=0
cond_wait()
UNLOCK
Sleep
3
2
2
1
1
0
3
Thread 1
Thread 2
Thread 3
remain
wakeup
Lock
Remain !=0
cond_wait()
UNLOCK
Sleep
Sleep
Z
Z
Z
Sleep
Z
Z
Z
3

P3

Program A

Right

```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  free(vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i, *ptr;
  for (i = 0; i < 2; i++) {
    ptr = malloc(sizeof(int));
    *ptr = i;
    pthread_create(&tid[i], 0, foo, ptr);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

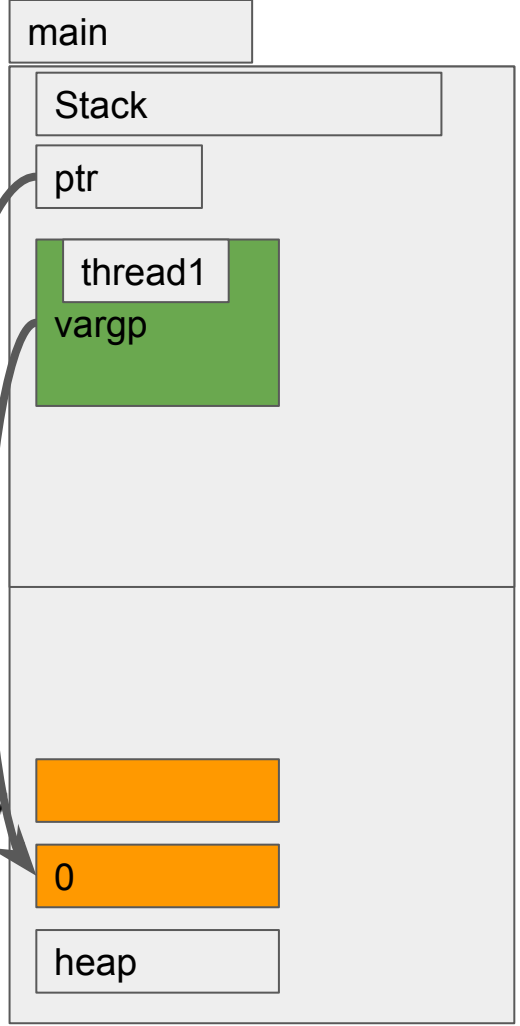


P3

Program A

Right

```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  free(vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i, *ptr;
  for (i = 0; i < 2; i++) {
    ptr = malloc(sizeof(int));
    *ptr = i;
    pthread_create(&tid[i], 0, foo, ptr);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

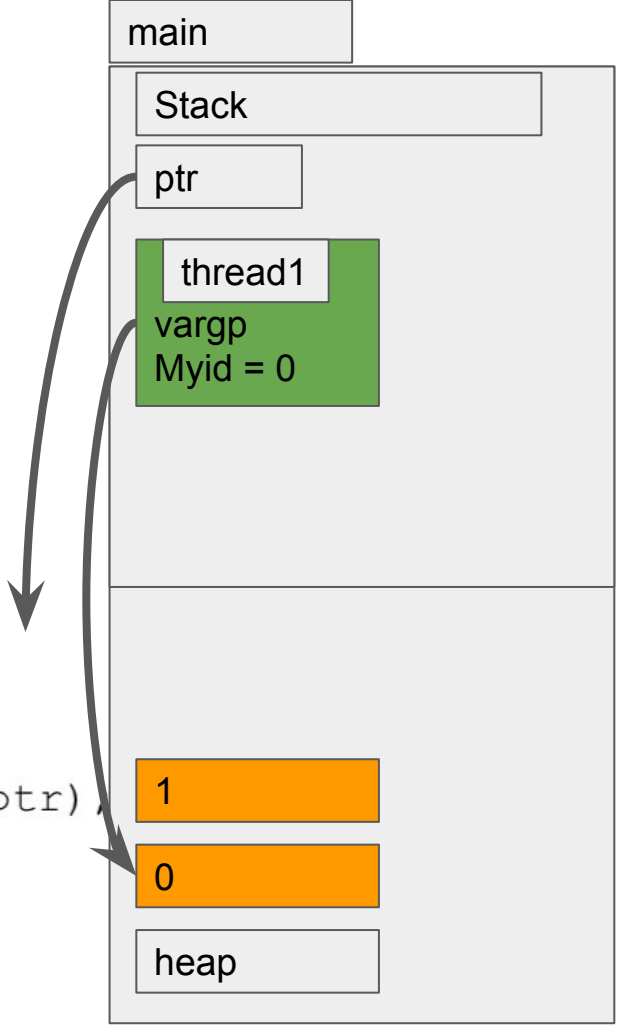


P3

Program A

Right

```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  free(vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i, *ptr;
  for (i = 0; i < 2; i++) {
    ptr = malloc(sizeof(int));
    *ptr = i;
    pthread_create(&tid[i], 0, foo, ptr);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

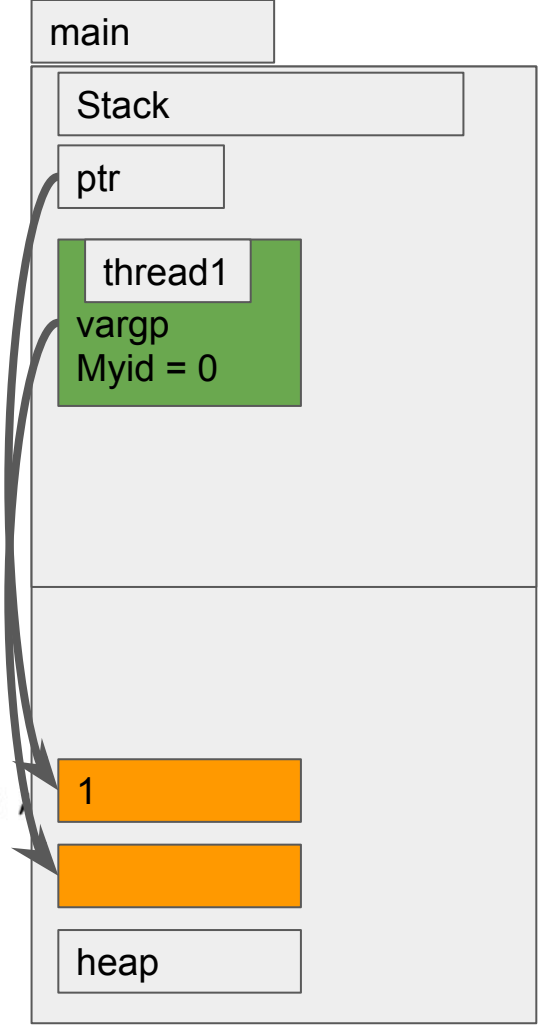


P3

Program A

Right

```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  free(vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i, *ptr;
  for (i = 0; i < 2; i++) {
    ptr = malloc(sizeof(int));
    *ptr = i;
    pthread_create(&tid[i], 0, foo, ptr);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

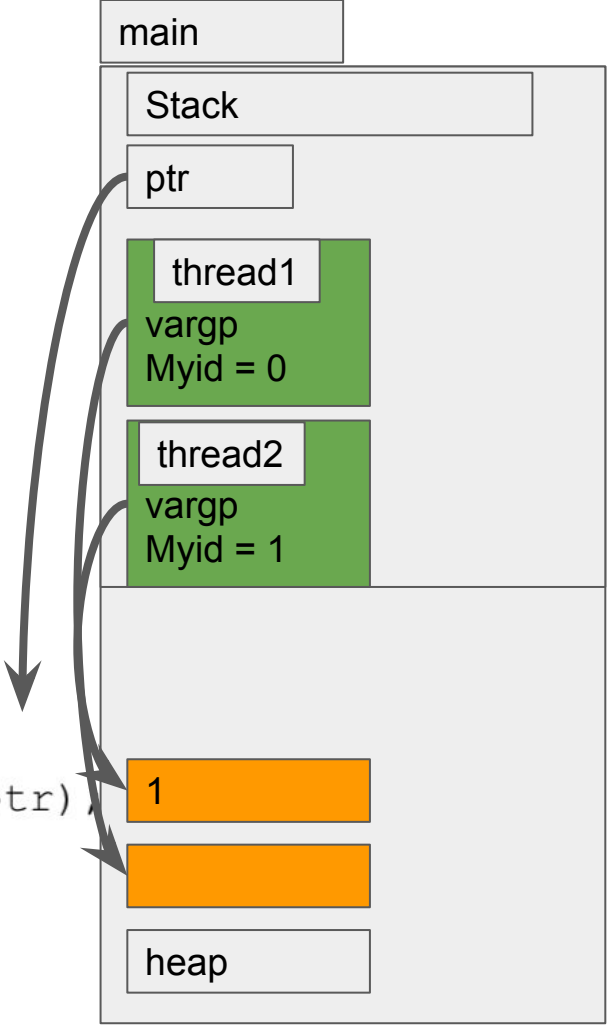


P3

Program A

Right

```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  free(vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i, *ptr;
  for (i = 0; i < 2; i++) {
    ptr = malloc(sizeof(int));
    *ptr = i;
    pthread_create(&tid[i], 0, foo, ptr);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

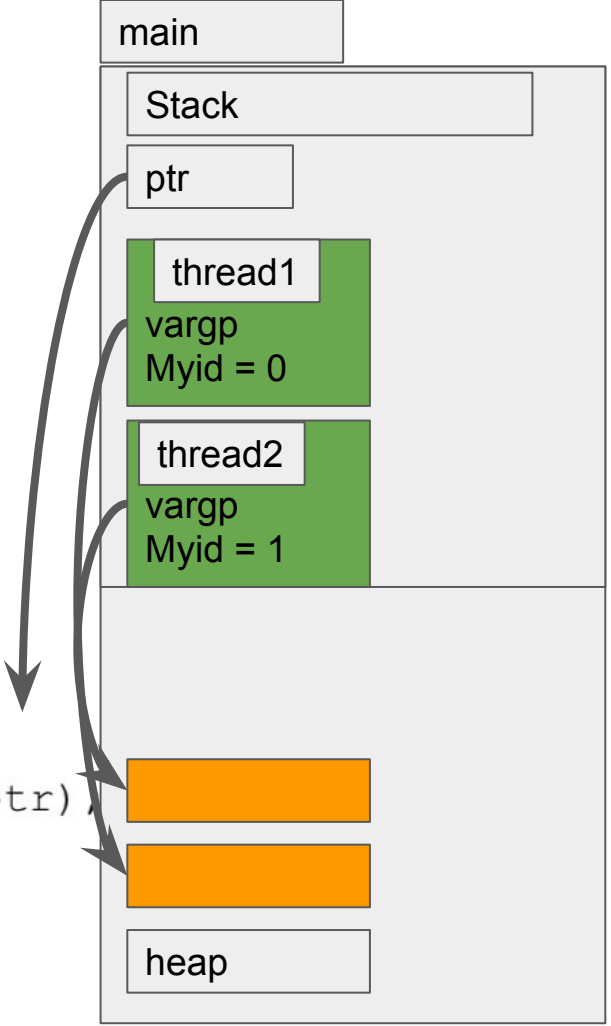


P3

Program A

Right

```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  free(vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i, *ptr;
  for (i = 0; i < 2; i++) {
    ptr = malloc(sizeof(int));
    *ptr = i;
    pthread_create(&tid[i], 0, foo, ptr);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

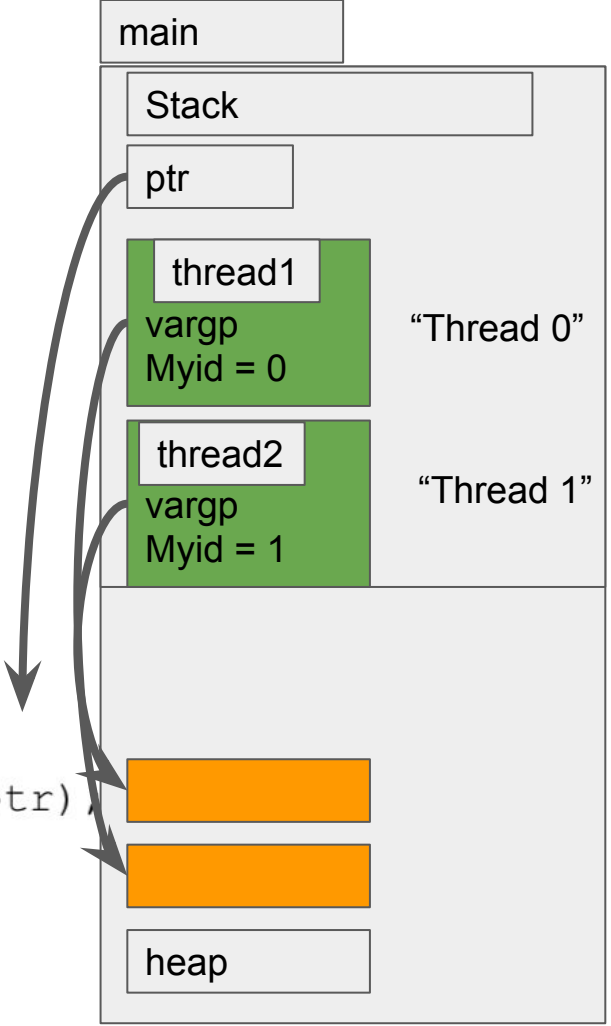


P3

Program A

Right

```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  free(vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i, *ptr;
  for (i = 0; i < 2; i++) {
    ptr = malloc(sizeof(int));
    *ptr = i;
    pthread_create(&tid[i], 0, foo, ptr);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

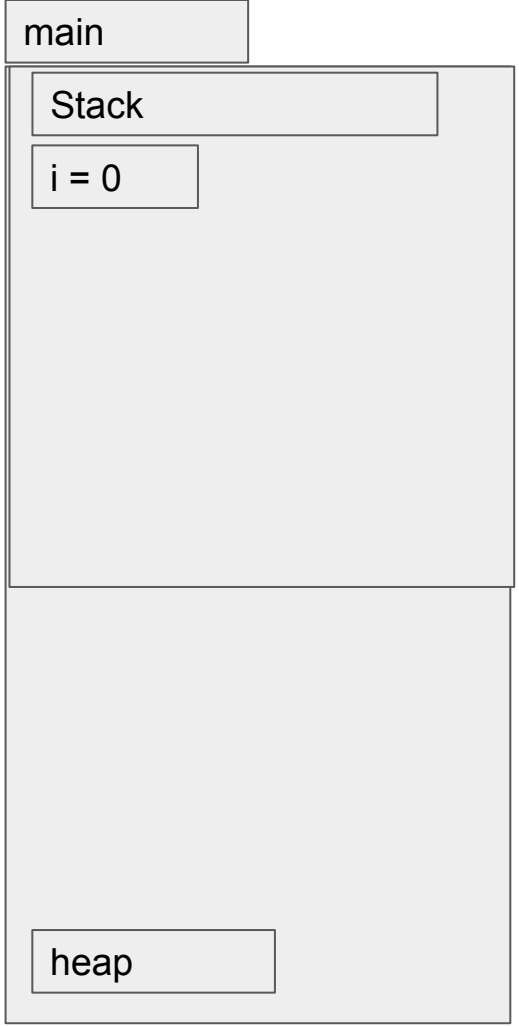


P3

Wrong

Program B

```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], NULL, foo, &i);
  }
  pthread_join(tid[0], NULL);
  pthread_join(tid[1], NULL);
}
```



P3

Wrong

Program B

```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], NULL, foo, &i);
  }
  pthread_join(tid[0], NULL);
  pthread_join(tid[1], NULL);
}
```

P3

Wrong

Program B

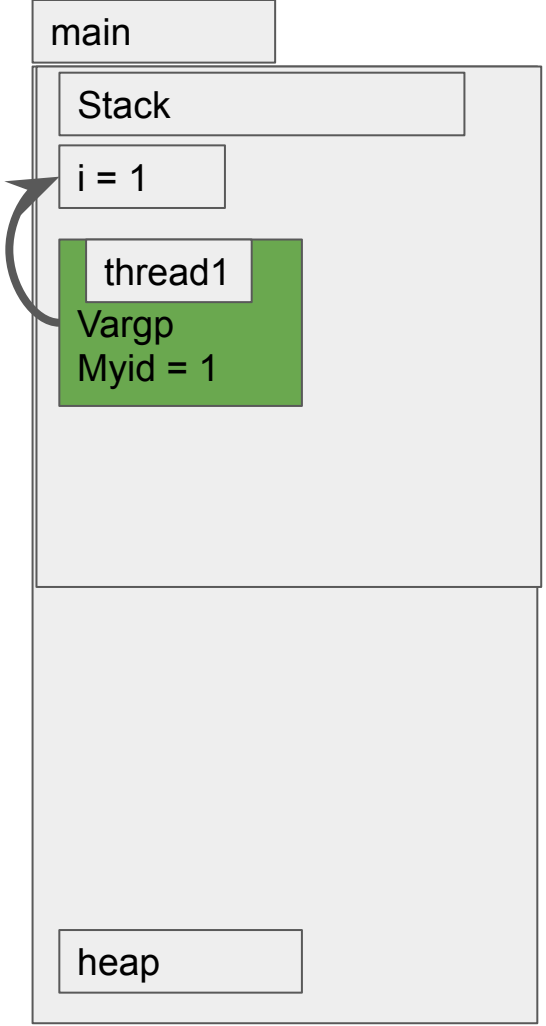
```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], NULL, foo, &i);
  }
  pthread_join(tid[0], NULL);
  pthread_join(tid[1], NULL);
}
```

P3

Wrong

Program B

```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], NULL, foo, &i);
  }
  pthread_join(tid[0], NULL);
  pthread_join(tid[1], NULL);
}
```

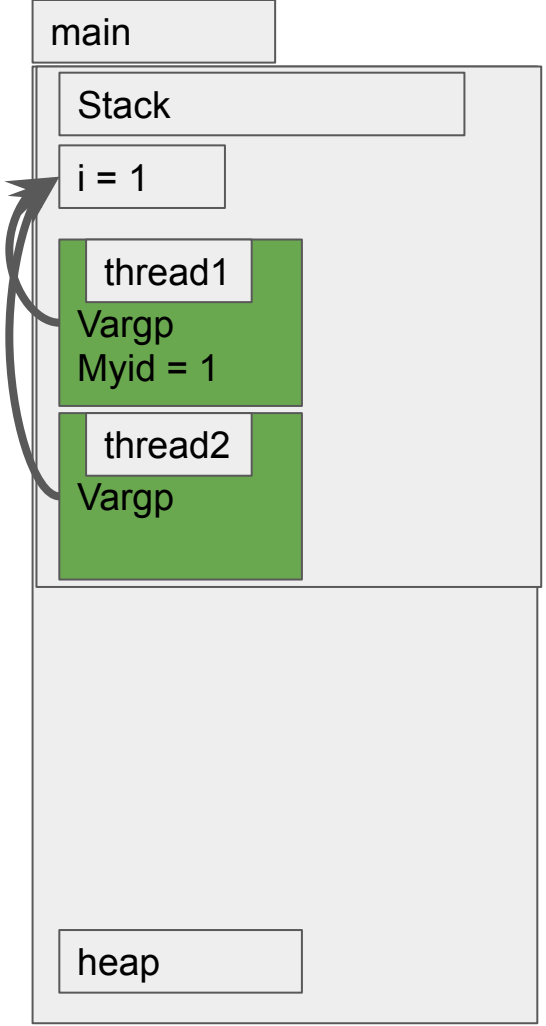


P3

Wrong

Program B

```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], NULL, foo, &i);
  }
  pthread_join(tid[0], NULL);
  pthread_join(tid[1], NULL);
}
```

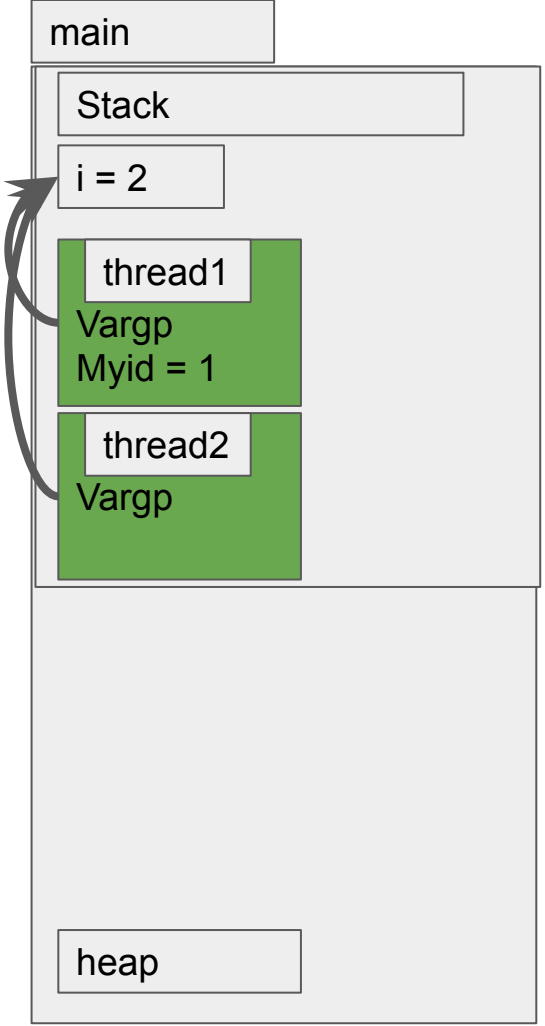


P3

Wrong

Program B

```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], NULL, foo, &i);
  }
  pthread_join(tid[0], NULL);
  pthread_join(tid[1], NULL);
}
```

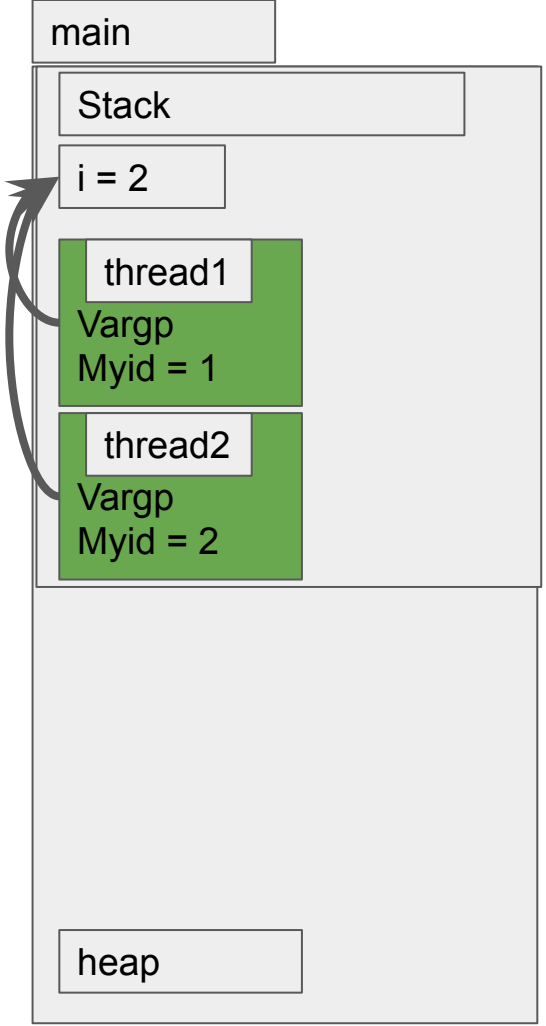


P3

Wrong

Program B

```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], NULL, foo, &i);
  }
  pthread_join(tid[0], NULL);
  pthread_join(tid[1], NULL);
}
```

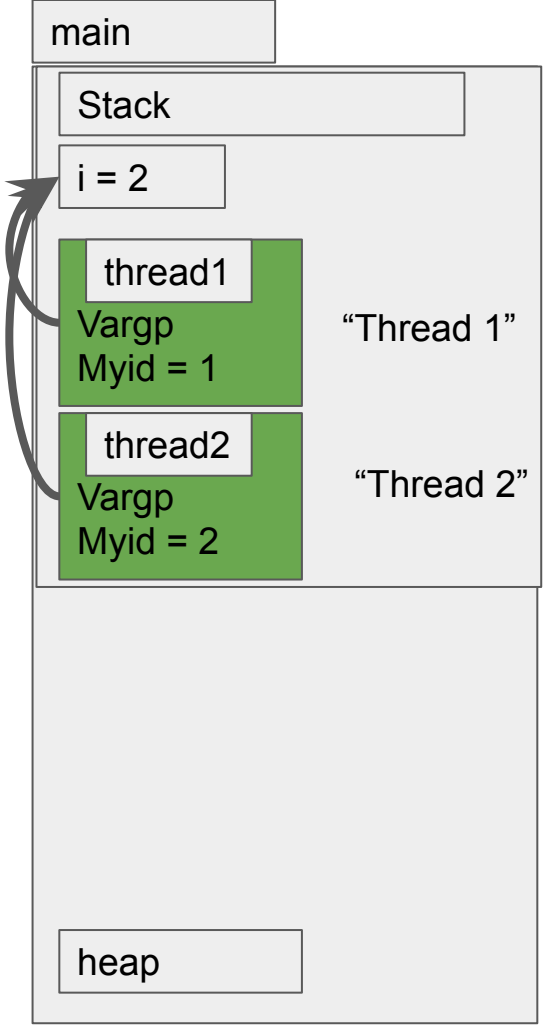


P3

Wrong

Program B

```
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], NULL, foo, &i);
  }
  pthread_join(tid[0], NULL);
  pthread_join(tid[1], NULL);
}
```

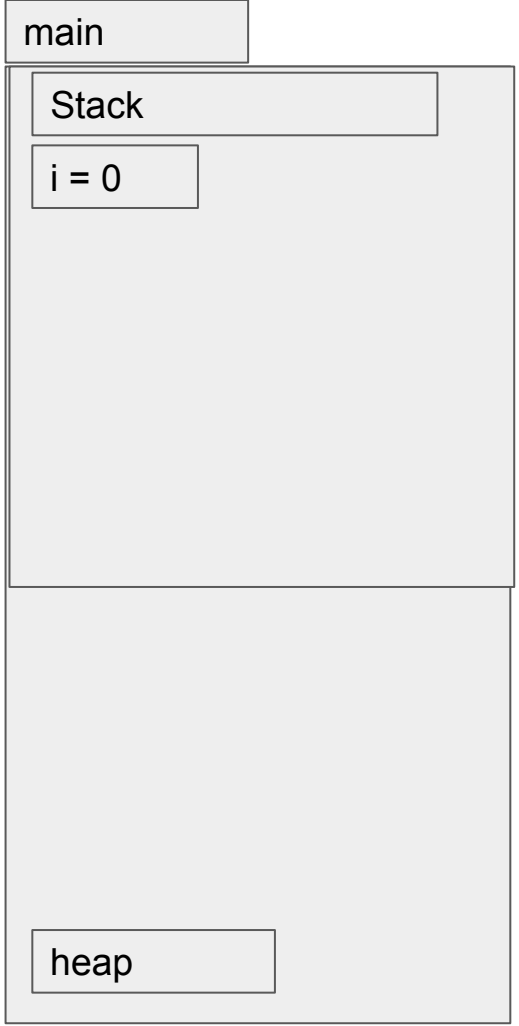


P3

Right

Program C

```
void *foo(void *vargp) {
  int myid;
  myid = (int)vargp;
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  for (i = 0; i < 2; i++)
    pthread_create(&tid[i], 0, foo, i);
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

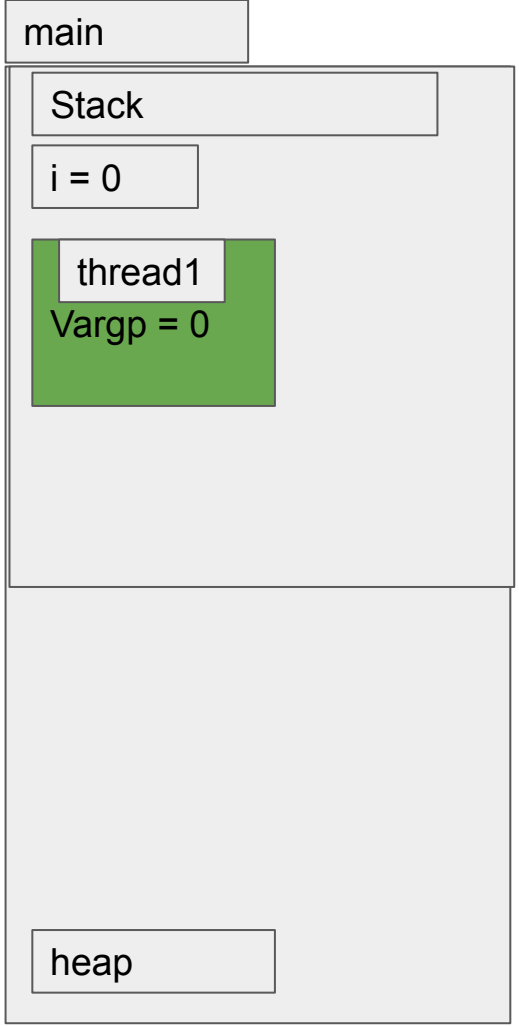


P3

Right

Program C

```
void *foo(void *vargp) {
  int myid;
  myid = (int)vargp;
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  for (i = 0; i < 2; i++)
    pthread_create(&tid[i], 0, foo, i);
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

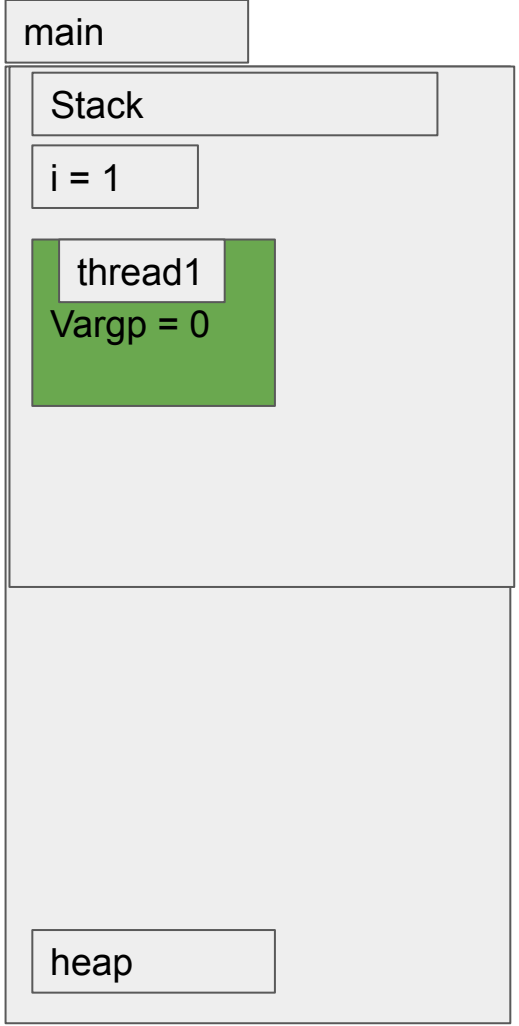


P3

Right

Program C

```
void *foo(void *vargp) {
  int myid;
  myid = (int)vargp;
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  for (i = 0; i < 2; i++)
    pthread_create(&tid[i], 0, foo, i);
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

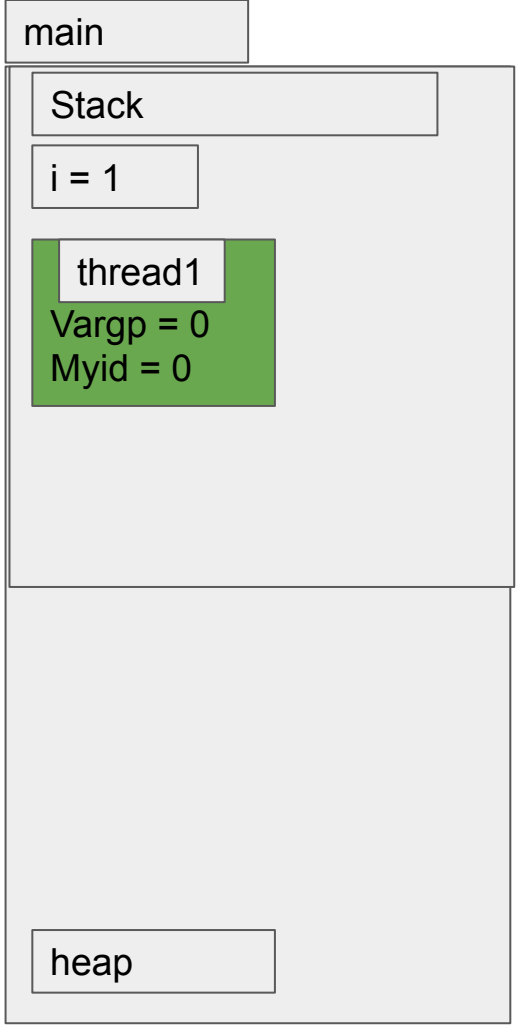


P3

Right

Program C

```
void *foo(void *vargp) {
  int myid;
  myid = (int)vargp;
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  for (i = 0; i < 2; i++)
    pthread_create(&tid[i], 0, foo, i);
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

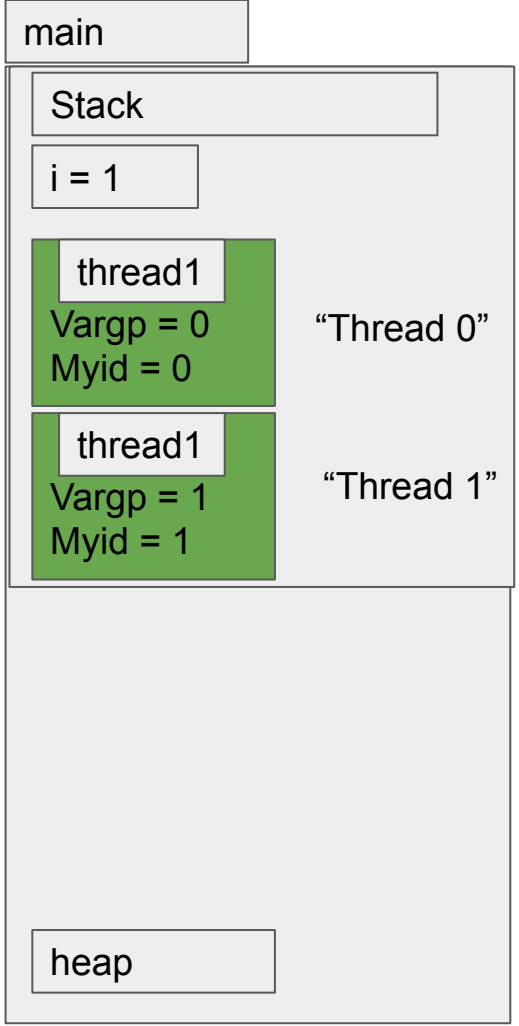


P3

Right

Program C

```
void *foo(void *vargp) {
  int myid;
  myid = (int)vargp;
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  for (i = 0; i < 2; i++)
    pthread_create(&tid[i], 0, foo, i);
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

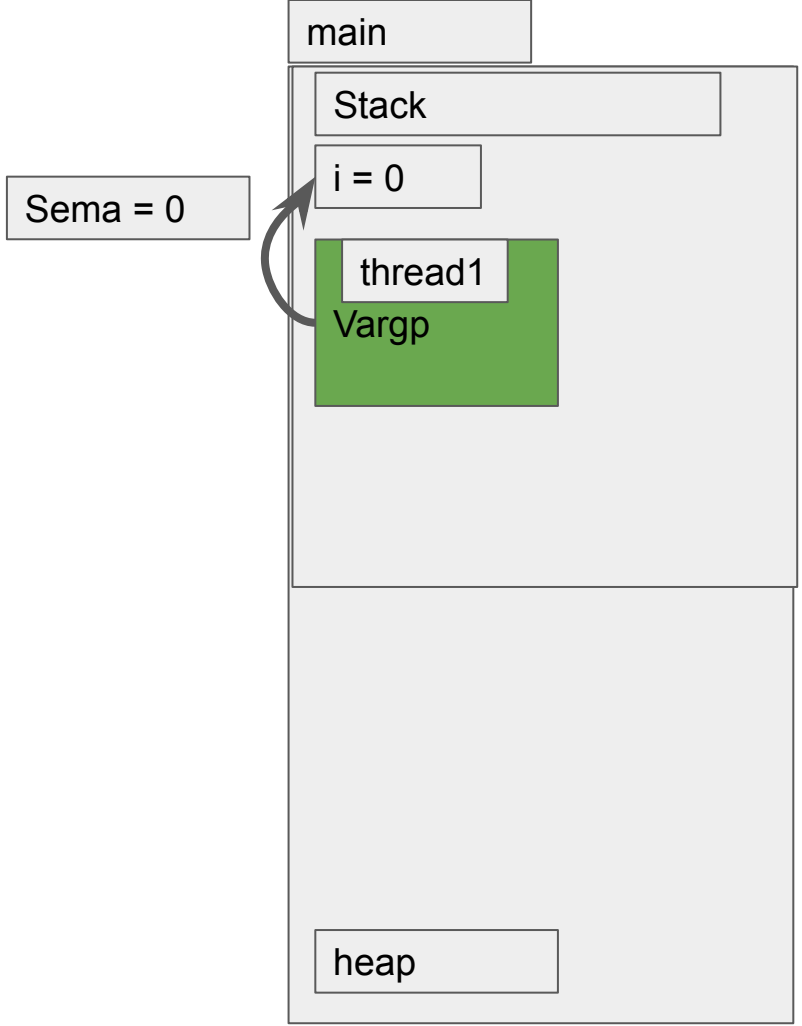


P3

Right

Program D

```
sem_t s; /* Semaphore s */
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  sem_post(&s);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  sem_init(&s, 0, 0); /* S=0 INITIALLY */
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], 0, foo, &i);
    sem_wait(&s);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

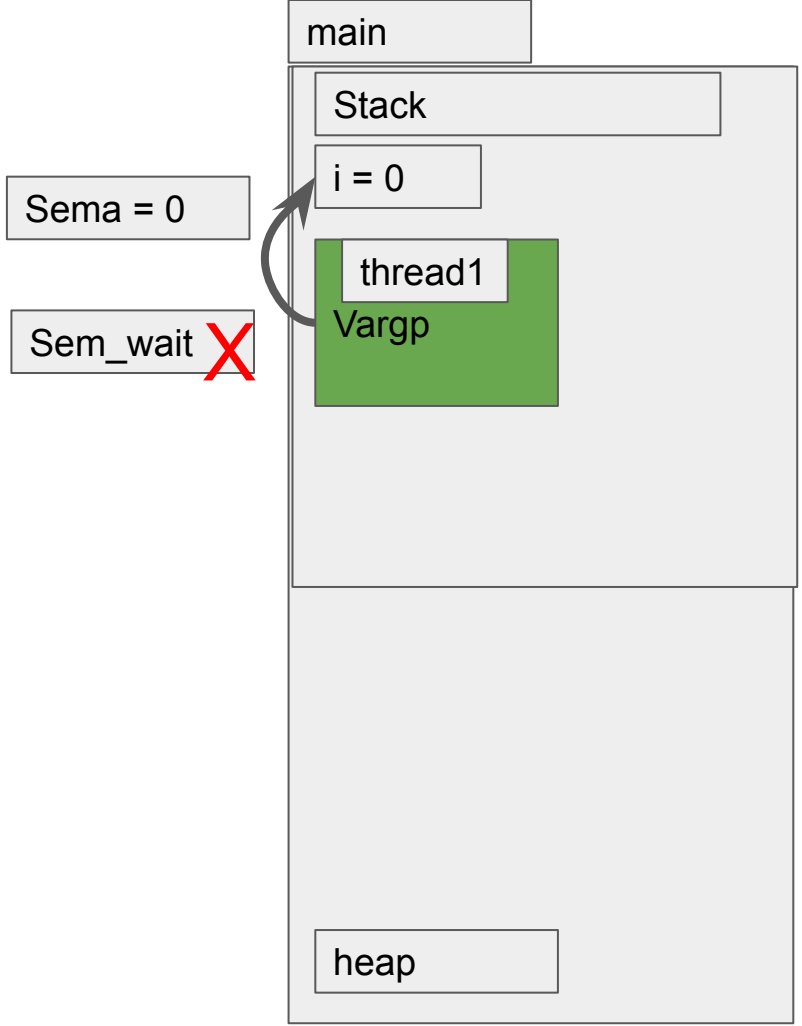


P3

Right

Program D

```
sem_t s; /* Semaphore s */
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  sem_post(&s);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  sem_init(&s, 0, 0); /* S=0 INITIALLY */
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], 0, foo, &i);
    sem_wait(&s);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

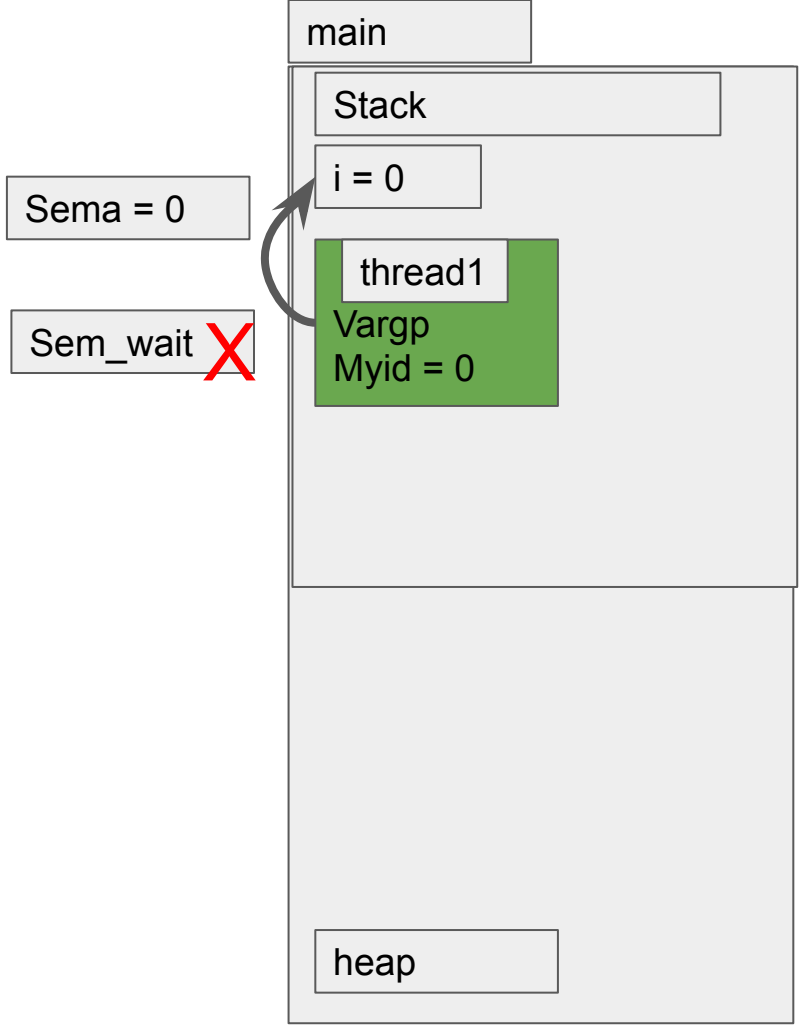


P3

Right

Program D

```
sem_t s; /* Semaphore s */
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  sem_post(&s);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  sem_init(&s, 0, 0); /* S=0 INITIALLY */
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], 0, foo, &i);
    sem_wait(&s);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

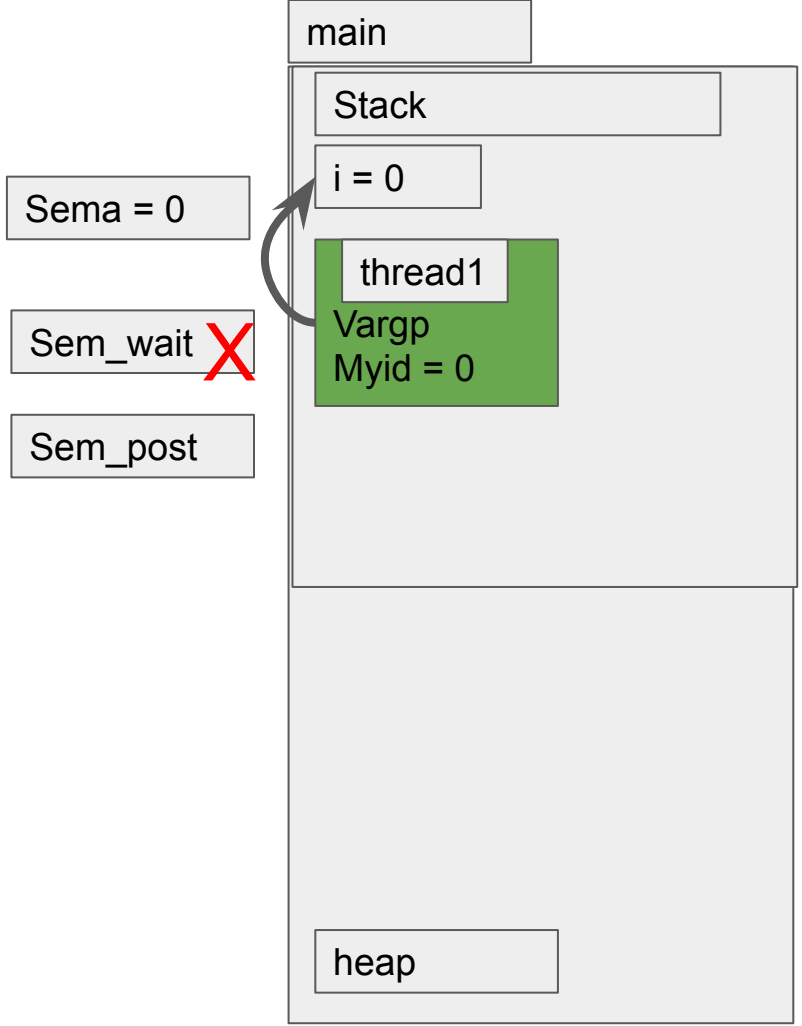


P3

Right

Program D

```
sem_t s; /* Semaphore s */
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  sem_post(&s);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  sem_init(&s, 0, 0); /* S=0 INITIALLY */
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], 0, foo, &i);
    sem_wait(&s);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

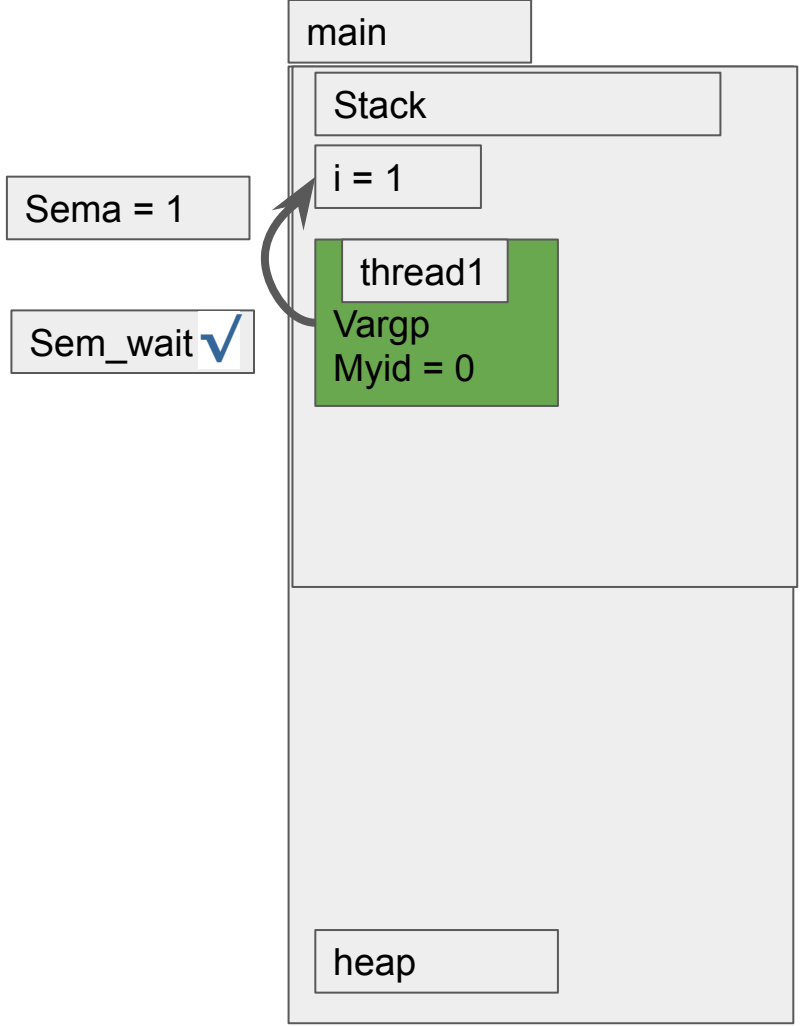


P3

Right

Program D

```
sem_t s; /* Semaphore s */
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  sem_post(&s);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  sem_init(&s, 0, 0); /* S=0 INITIALLY */
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], 0, foo, &i);
    sem_wait(&s);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

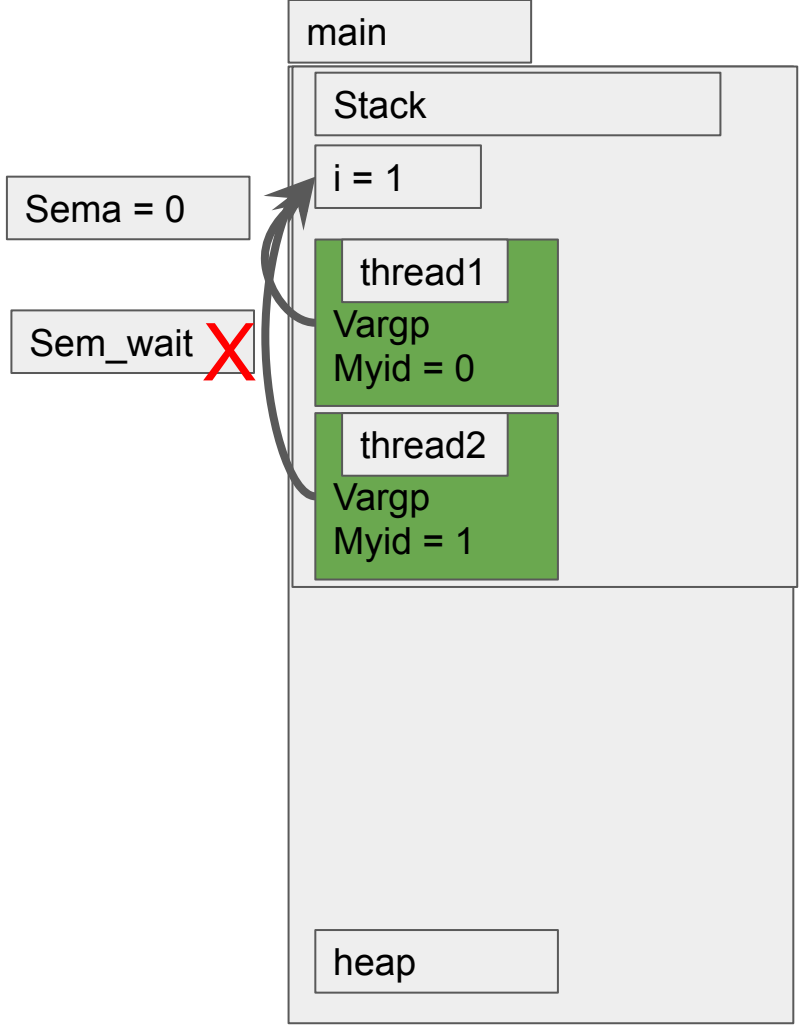


P3

Program D

```
sem_t s; /* Semaphore s */
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  sem_post(&s);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  sem_init(&s, 0, 0); /* S=0 INITIALLY */
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], 0, foo, &i);
    sem_wait(&s);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

Right

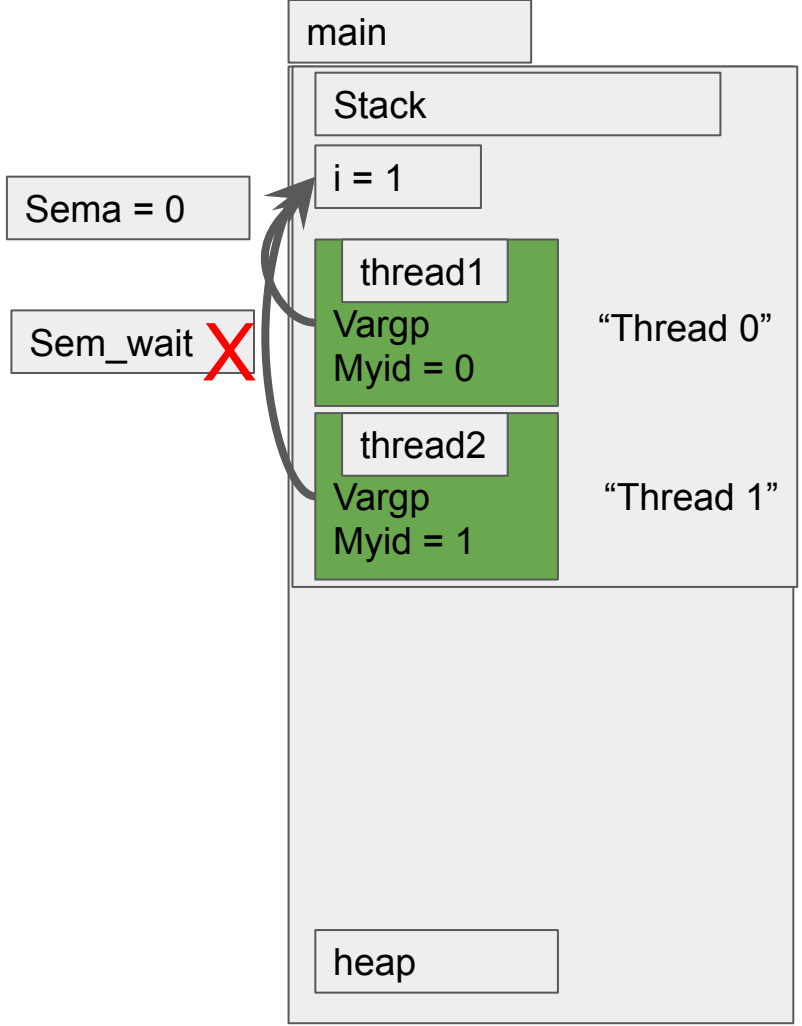


P3

Right

Program D

```
sem_t s; /* Semaphore s */
void *foo(void *vargp) {
  int myid;
  myid = *((int *)vargp);
  sem_post(&s);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  sem_init(&s, 0, 0); /* S=0 INITIALLY */
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], 0, foo, &i);
    sem_wait(&s);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

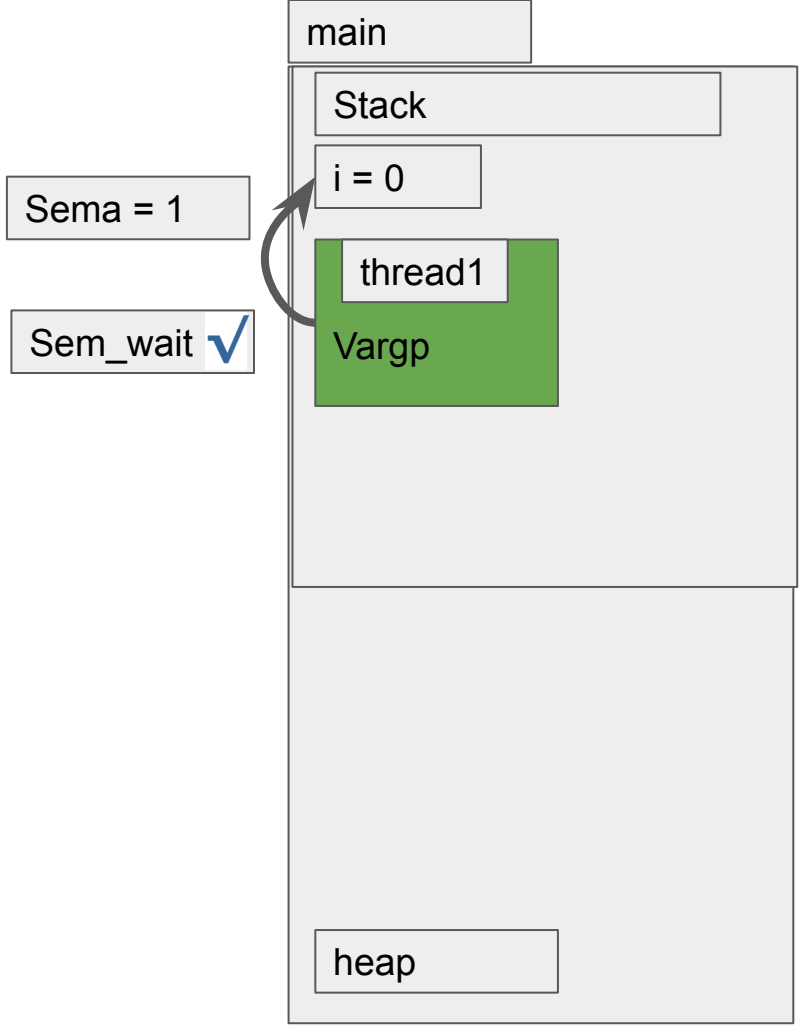


P3

WRONG

Program E

```
sem_t s; /* Semaphore s */
void *foo(void *vargp) {
  int myid;
  sem_wait(&s);
  myid = *((int *)vargp);
  sem_post(&s);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  sem_init(&s, 0, 1); /* S=1 INITIALLY */
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], 0, foo, &i);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

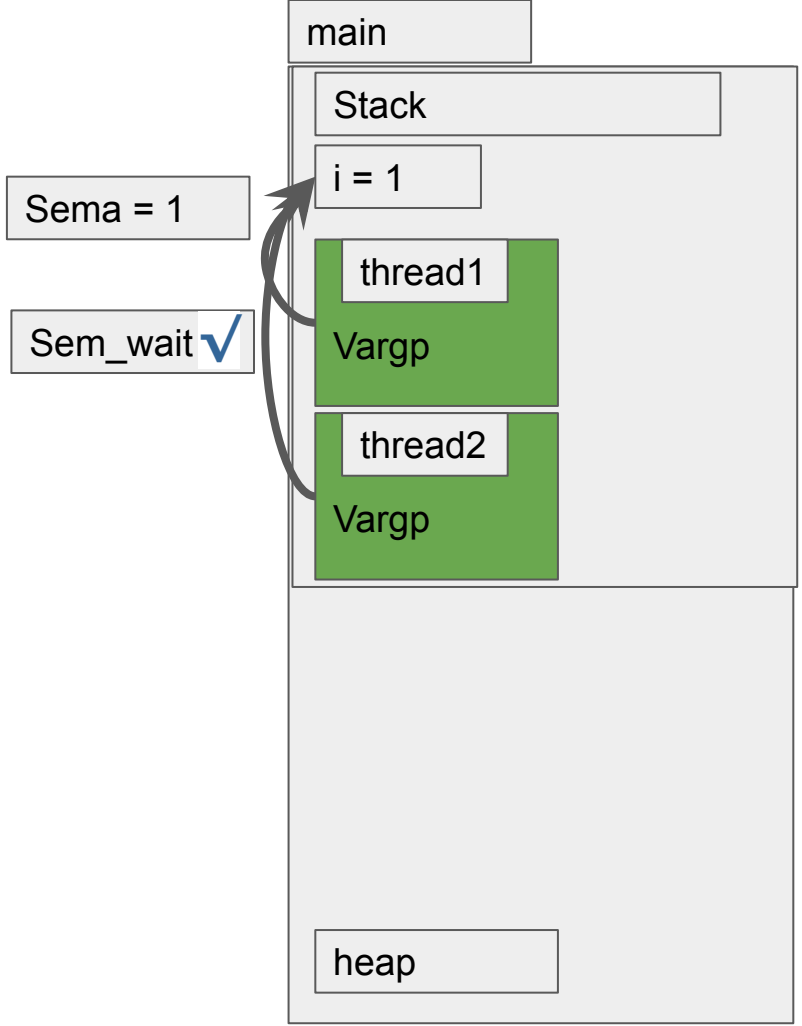


P3

WRONG

Program E

```
sem_t s; /* Semaphore s */
void *foo(void *vargp) {
  int myid;
  sem_wait(&s);
  myid = *((int *)vargp);
  sem_post(&s);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  sem_init(&s, 0, 1); /* S=1 INITIALLY */
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], 0, foo, &i);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

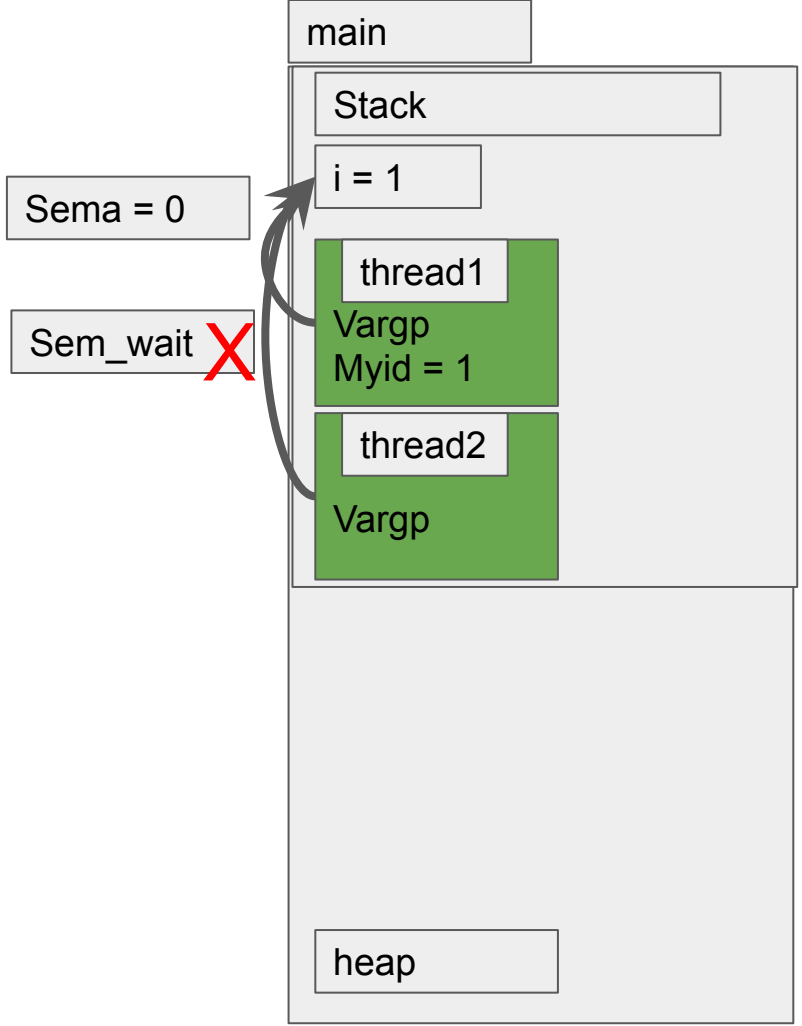


P3

WRONG

Program E

```
sem_t s; /* Semaphore s */
void *foo(void *vargp) {
  int myid;
  sem_wait(&s);
  myid = *((int *)vargp);
  sem_post(&s);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  sem_init(&s, 0, 1); /* S=1 INITIALLY */
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], 0, foo, &i);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

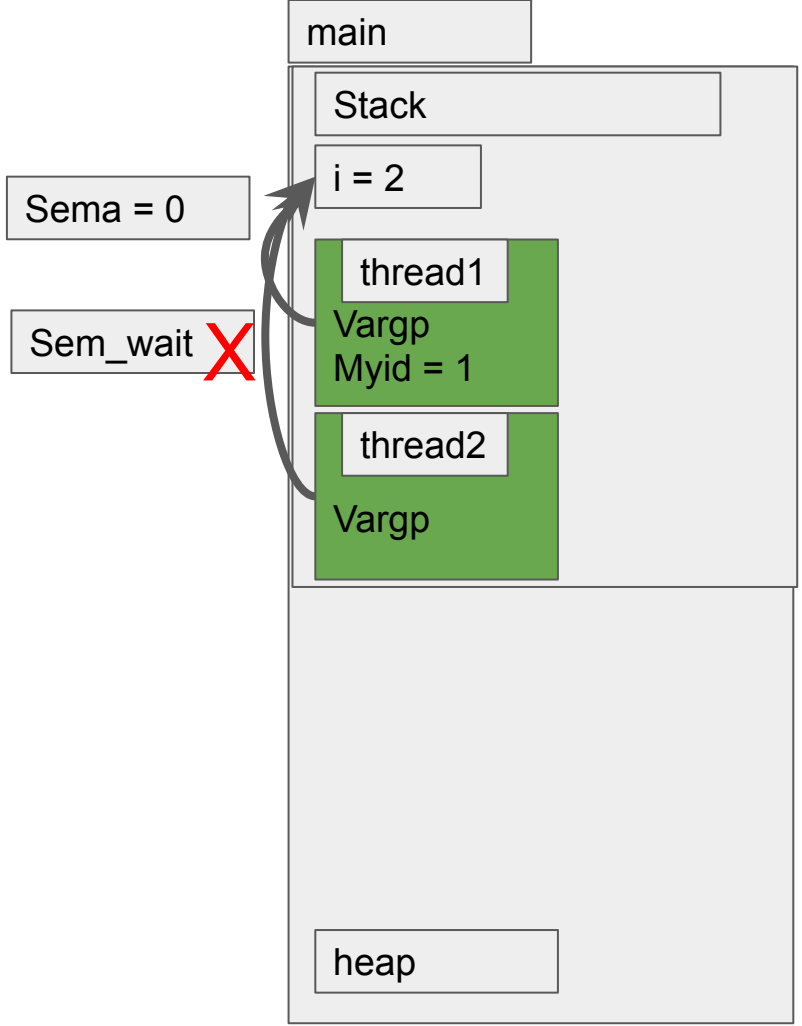


P3

WRONG

Program E

```
sem_t s; /* Semaphore s */
void *foo(void *vargp) {
  int myid;
  sem_wait(&s);
  myid = *((int *)vargp);
  sem_post(&s);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  sem_init(&s, 0, 1); /* S=1 INITIALLY */
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], 0, foo, &i);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

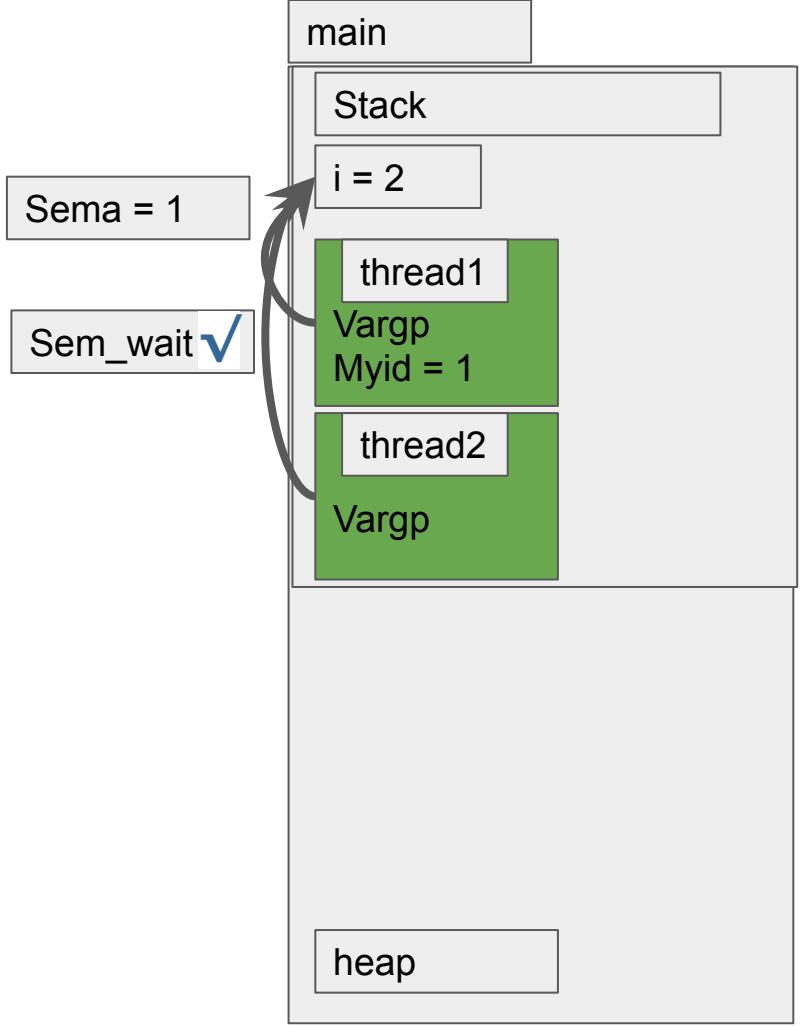


P3

WRONG

Program E

```
sem_t s; /* Semaphore s */
void *foo(void *vargp) {
  int myid;
  sem_wait(&s);
  myid = *((int *)vargp);
  sem_post(&s);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  sem_init(&s, 0, 1); /* S=1 INITIALLY */
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], 0, foo, &i);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

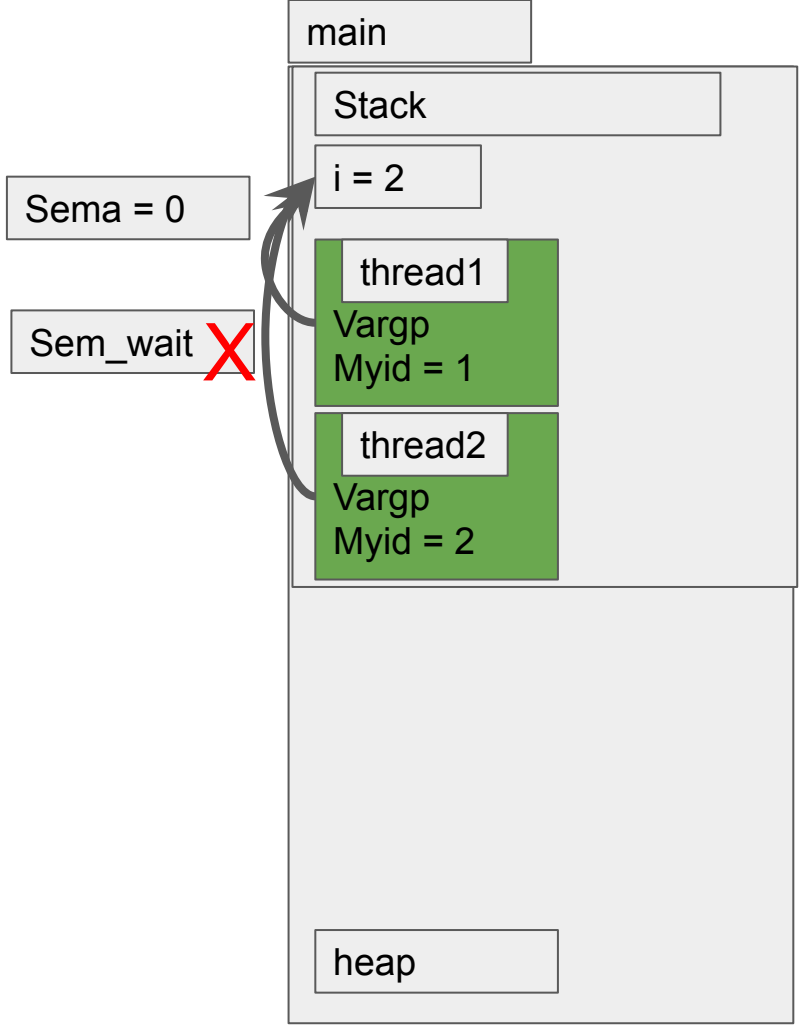


P3

WRONG

Program E

```
sem_t s; /* Semaphore s */
void *foo(void *vargp) {
  int myid;
  sem_wait(&s);
  myid = *((int *)vargp);
  sem_post(&s);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  sem_init(&s, 0, 1); /* S=1 INITIALLY */
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], 0, foo, &i);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```

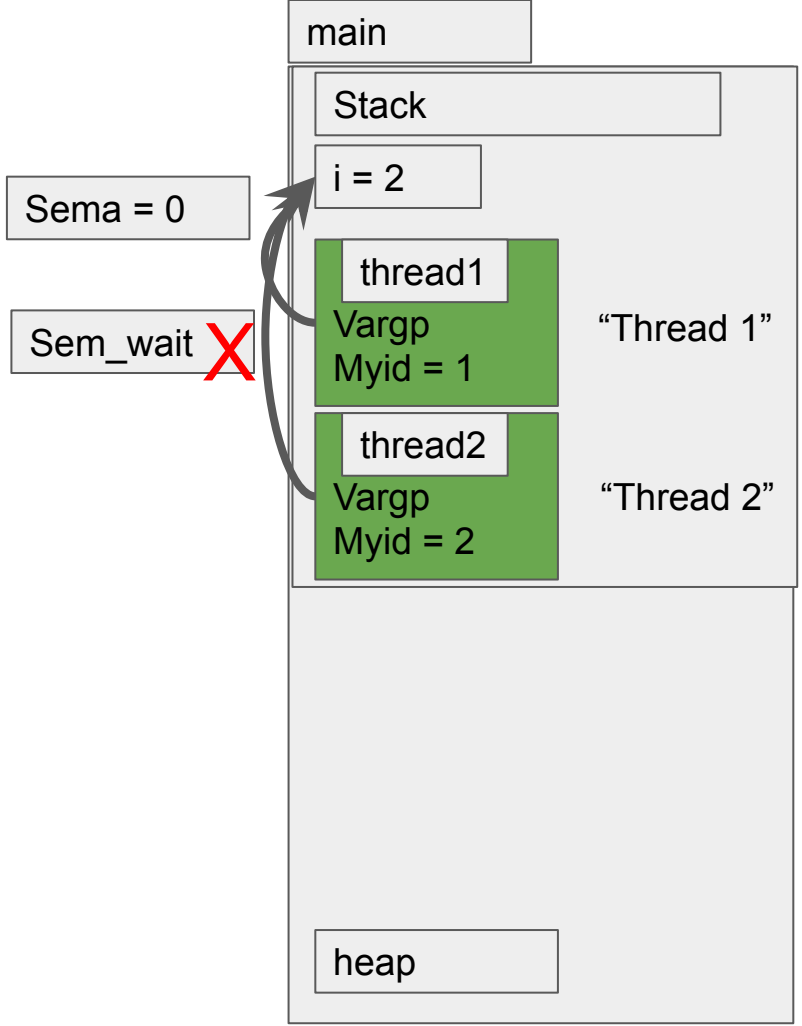


P3

WRONG

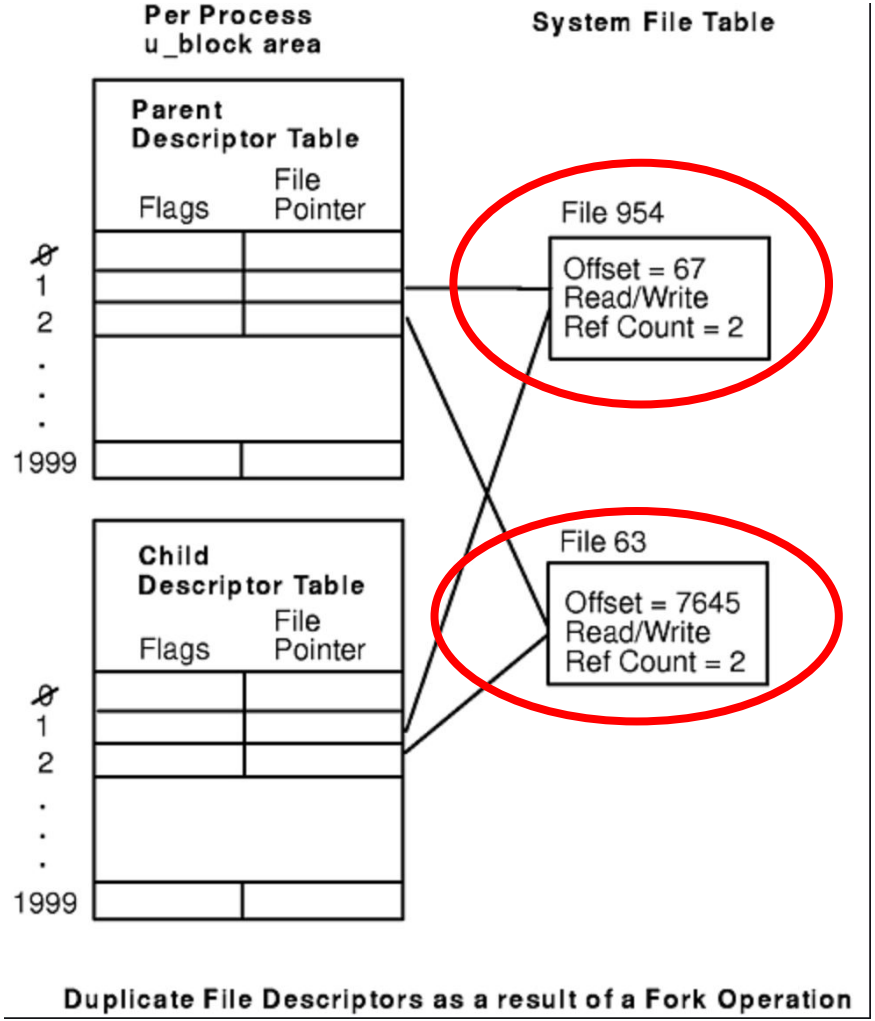
Program E

```
sem_t s; /* Semaphore s */
void *foo(void *vargp) {
  int myid;
  sem_wait(&s);
  myid = *((int *)vargp);
  sem_post(&s);
  printf("Thread %d\n", myid);
}

int main() {
  pthread_t tid[2];
  int i;
  sem_init(&s, 0, 1); /* S=1 INITIALLY */
  for (i = 0; i < 2; i++) {
    pthread_create(&tid[i], 0, foo, &i);
  }
  pthread_join(tid[0], 0);
  pthread_join(tid[1], 0);
}
```



- Worksheet
- quizizz
- Fill up midterm review form
- Code review
- Coding trick & Style workshop

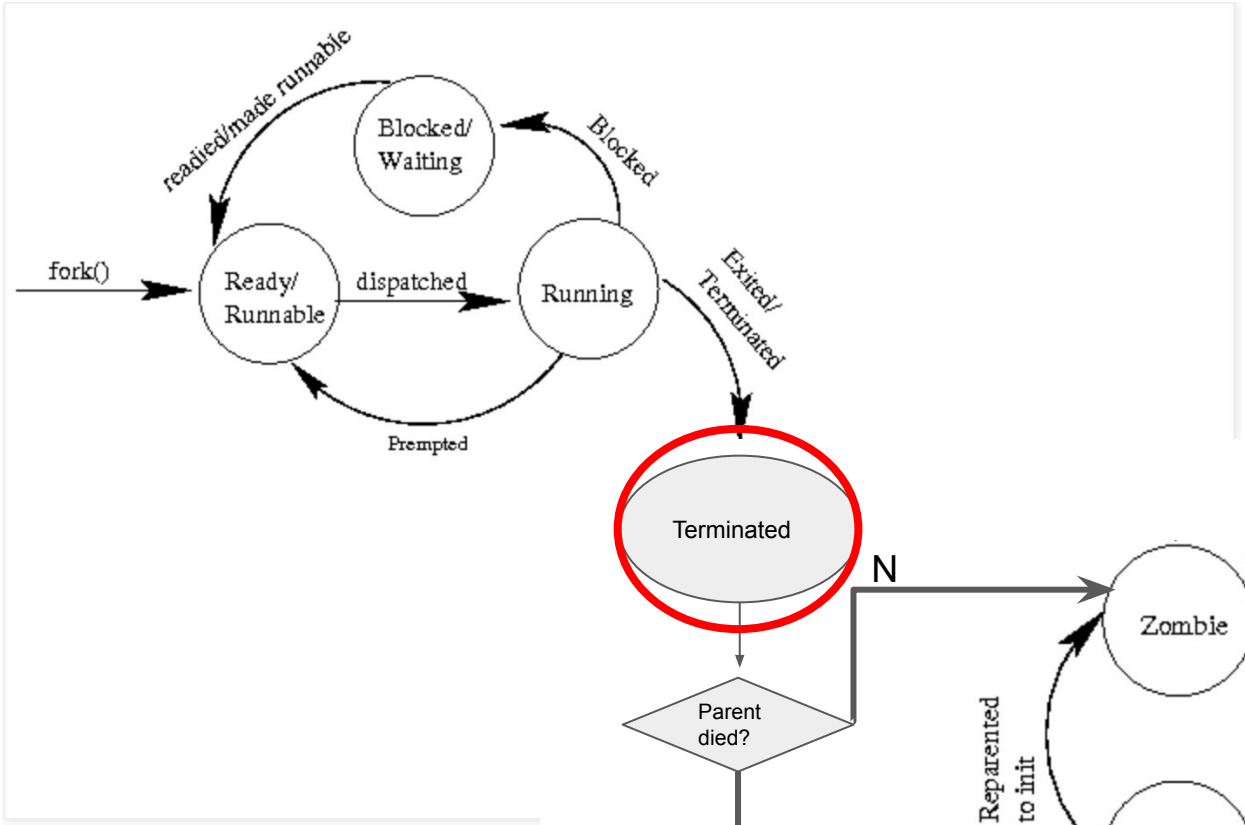
Q3



Duplicate File Descriptors as a result of a Fork Operation

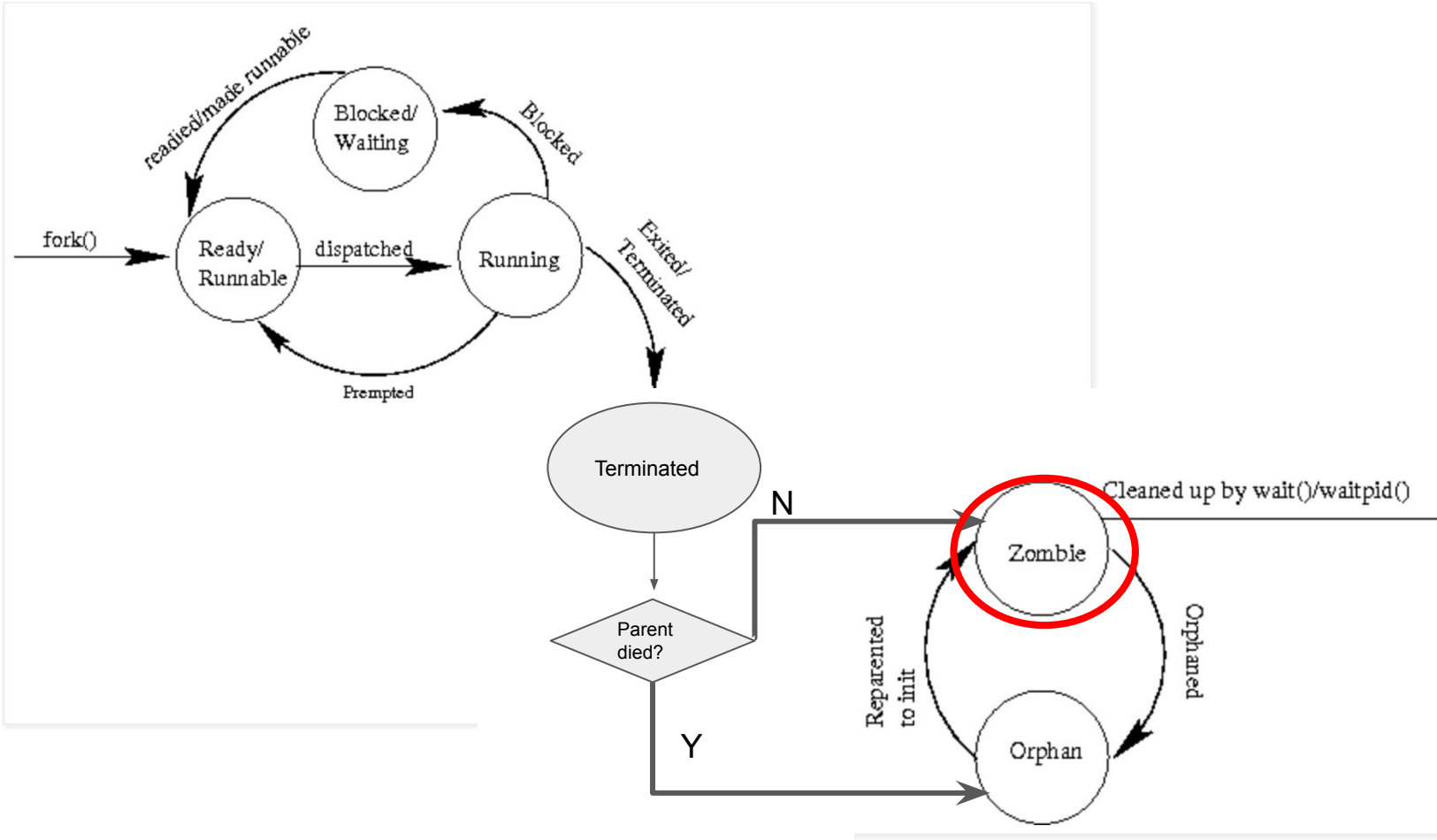
https://sites.ualberta.ca/dept/chemeng/AIX-43/share/man/info/C/a_doc_lib/aixprggd/genprogc/fdescript.htm

Q5



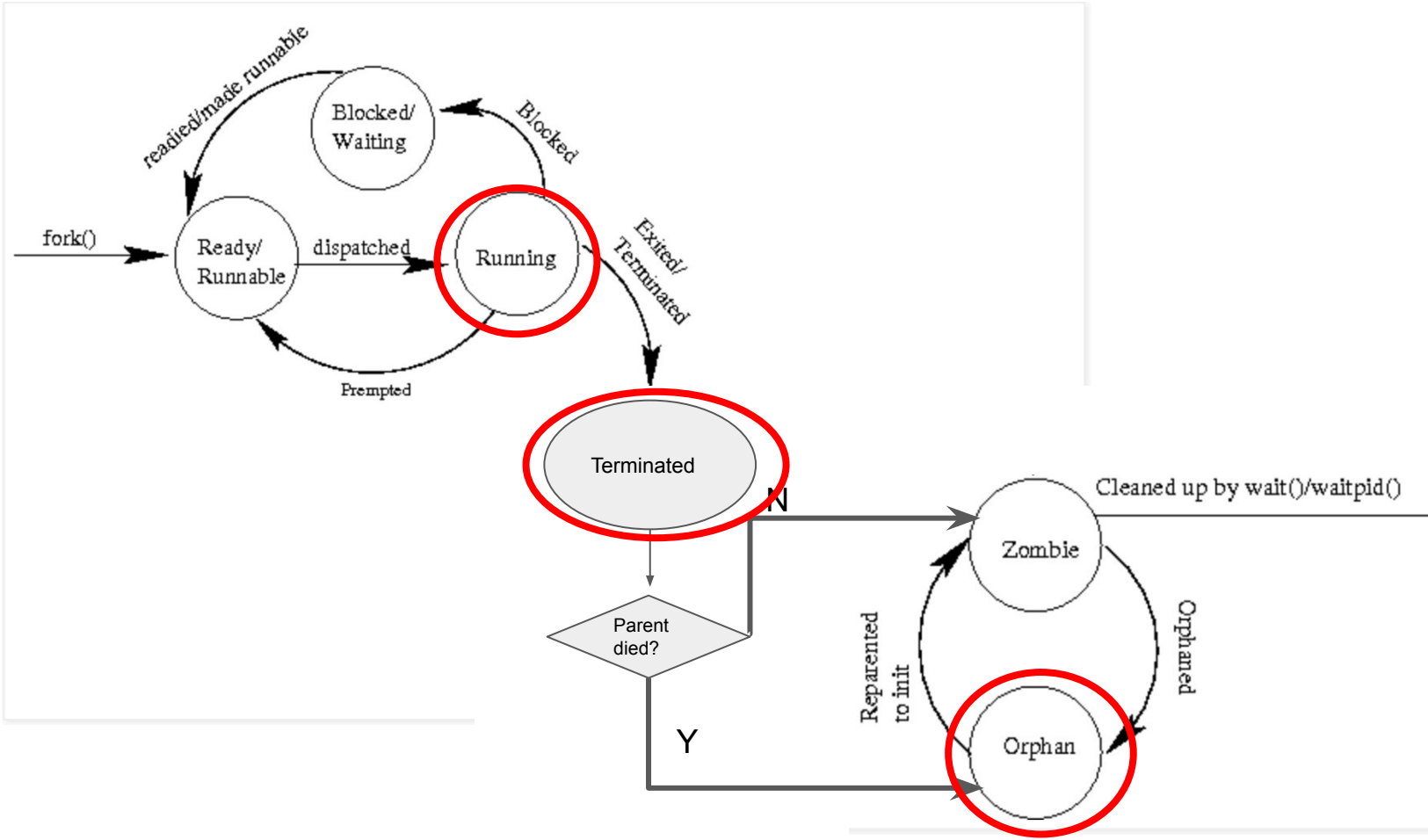
http://rafaxu.blogspot.com/2014/08/a-good-picture-about-linux-process-life.html

Q6



http://rafaxu.blogspot.com/2014/08/a-good-picture-about-linux-process-life.html

Q7



http://rafaxu.blogspot.com/2014/08/a-good-picture-about-linux-process-life.html

- Worksheet
- quizizz
- Code review
- Coding Trick & Style workshop
- Fill up midterm review form

- Discuss your Best code and Worst Code with your neighbors
  - comments in code
  - writing modular code
  - good naming conventions
  - proper spacing and indentation, etc.

- Worksheet
- quizizz
- Code review
- Coding Trick & Style workshop
- Fill up midterm review form

- Worksheet
- quizizz
- Code review
- Coding trick & Style workshop
- Fill up midterm review form