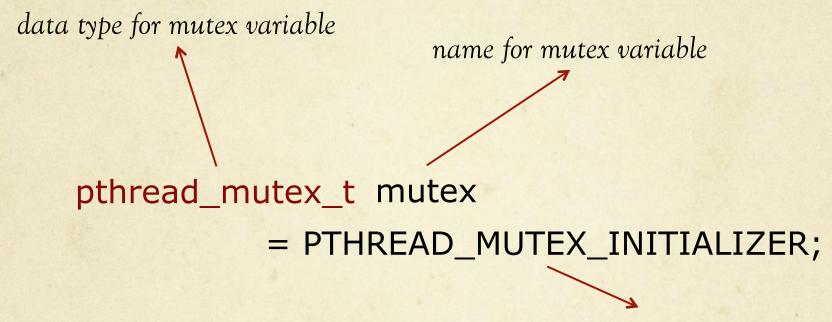
Pthread mutex



macro to initialize mutex variable with default attributes

Pthread mutex

```
int pthread_mutex_lock(pthread_mutex_t* p_mutex);

status

address of mutex variable

int pthread_mutex_unlock(pthread_mutex_t* p_mutex);
```

```
#include <pthread.h>
#include <iostream>
using namespace std;
int account_balance;
```

Sample program - 6

int account_balance; pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;

```
void* deposit(void* arg) {
  int amount = *((int*)arg);
  cout << "Depositing $" << amount << endl;
  pthread_mutex_lock(&mutex);
  account_balance += amount;
  pthread_mutex_unlock(&mutex);
  pthread_exit(NULL);
}</pre>
```

```
void* withdraw(void* arg) {
  int amount = *((int*)arg);
  cout << "Withdrawing $" << amount << endl;
  pthread_mutex_lock(&mutex);
  account_balance -= amount;
  pthread_mutex_unlock(&mutex);
  pthread_exit(NULL);
}</pre>
```

```
int main () {
    pthread_t id1, id2; int d, w; int* pa;
    account_balance = 100;
    cout << "Initial balance: $" << account_balance << endl;
    d = 50; pa = &d;
    pthread_create(&id1, NULL, deposit, (void *)pa);
    w = 60; pa = &w;
    pthread_create(&id2, NULL, withdraw, (void *)pa);
    pthread_join(id1, NULL);
    pthread_join(id2, NULL);
    cout << "Final balance: $" << account_balance << endl;
    pthread_exit(NULL);
}</pre>
```