Thread
Create, Terminate

Thread Creation
The pthread_create function starts a new threads in the calling process.

```
int pthread_create(
     pthread_t* thread_id,
     const pthread_attr_t *attr,
     void *(*start_routine)(void*),
     void* arg);
```

- First argument, thread_id is used to store thread id.
- Second argument, attr is used to customize thread attributes. If attr is NULL, create thread with default attributes
- Third, the thread starts running at the address of start_routine function. This function take a single argument.
- Finally, arg is argument used to pass to start_routine function.

Note:
When two threads are created, no guarantee that which will run first.

Return Value
On success, return 0
On error, return failure number.

Thread Termination
A thread can exit on below conditions,
- It calls pthread_exit(retval), retval is the exit status value that is available to another thread in the same process that calls pthread_join.
- It returns from start routine function. This is equivalent to calling pthread_exit with the value supplied in the return statement.
- It is canceled. The exit code is set to PTHREAD_CANCELED.
- Its process terminate.

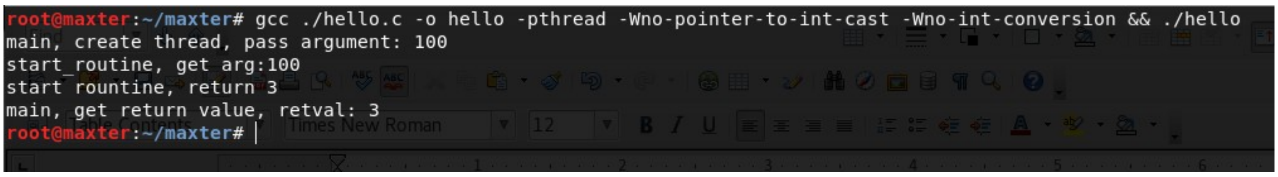
Example
Create thread, pass argument, get return value

```
#include <pthread.h>
#include <stdio.h>

void* start_routine(void* arg) {
    int n = (int)arg;
    printf("start_routine, get arg:%d\n", n);

    printf("start rountine, return 3\n");

    //return value
    pthread_exit((void*)3);
    // return (void*)3;
}

int main() {
    pthread_t thread_id;
    void* retval;//used to get thread return value

    printf("main, create thread, pass argument: 100\n");

    pthread_create(&thread_id, NULL, start_routine, 100);
    pthread_join(thread_id, &retval);

    int n = (int)retval;
    printf("main, get return value, retval: %d\n", retval);

    return 0;
}
```

Compile

```
gcc ./hello.c -o hello -pthread -Wno-pointer-to-int-cast -Wno-int-conversion
```

Result



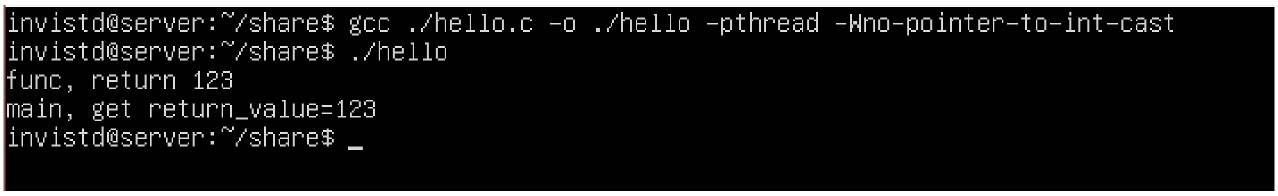
```
root@maxter:~/maxter# gcc ./hello.c -o hello -pthread -Wno-pointer-to-int-cast -Wno-int-conversion && ./hello
main, create thread, pass argument: 100
start_routine, get arg:100
start rountine, return 3
main, get return value, retval: 3
root@maxter:~/maxter# |
```

Example

Fetch thread return value (start routine function exit on return statement)

```
#include <pthread.h>
#include <stdio.h>

void* func(void* arg) {
    printf("func, return 123\n");
    return (void*)123;
}

int main(){
    pthread_t thread_1;
    void* return_value;

    pthread_create(&thread_1, NULL, func, NULL);
    pthread_join(thread_1, (void*)&return_value);

    printf("main, get return_value=%d\n", (int)return_value);

    return 1;
}
```

Result

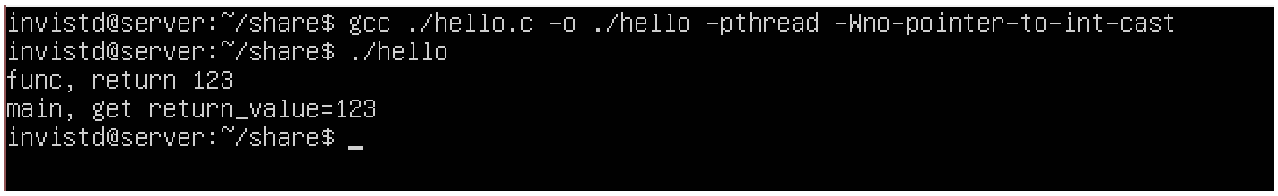


```
invistd@server:~/share$ gcc ./hello.c -o ./hello -pthread -Wno-pointer-to-int-cast
invistd@server:~/share$ ./hello
func, return 123
main, get return_value=123
invistd@server:~/share$ _
```

Example
Fetch thread return value ((start routine function exit on pthread_exit)

```
#include <pthread.h>
#include <stdio.h>

void* func(void* arg) {
    printf("func, return 123\n");
    // return (void*)123;
    pthread_exit((void*)123);
}

int main(){
    pthread_t thread_1;
    void* return_value;

    pthread_create(&thread_1, NULL, func, NULL);
    pthread_join(thread_1, (void*)&return_value);

    printf("main, get return_value=%d\n", (int)return_value);

    return 1;
}
```

Result



```
invistd@server:~/share$ gcc ./hello.c -o ./hello -pthread -Wno-pointer-to-int-cast
invistd@server:~/share$ ./hello
func, return 123
main, get return_value=123
invistd@server:~/share$ _
```

Exit start routine function by return statement vs pthread_exit
Cleanup-handler is execute if a thead terminated by pthread_exit. This does not happen if thread terminate by performing return statement in start routine function.

	Cleanup-handlers is executed
Exit by pthread_exit	**YES**
Exit by performing return statement	**NO**

Thread Cleanup Handler (continue...)
Cleanup-handler is a function that automatically executed when a thread is canceled.
It might be used to unlock mutex or release memory,... .