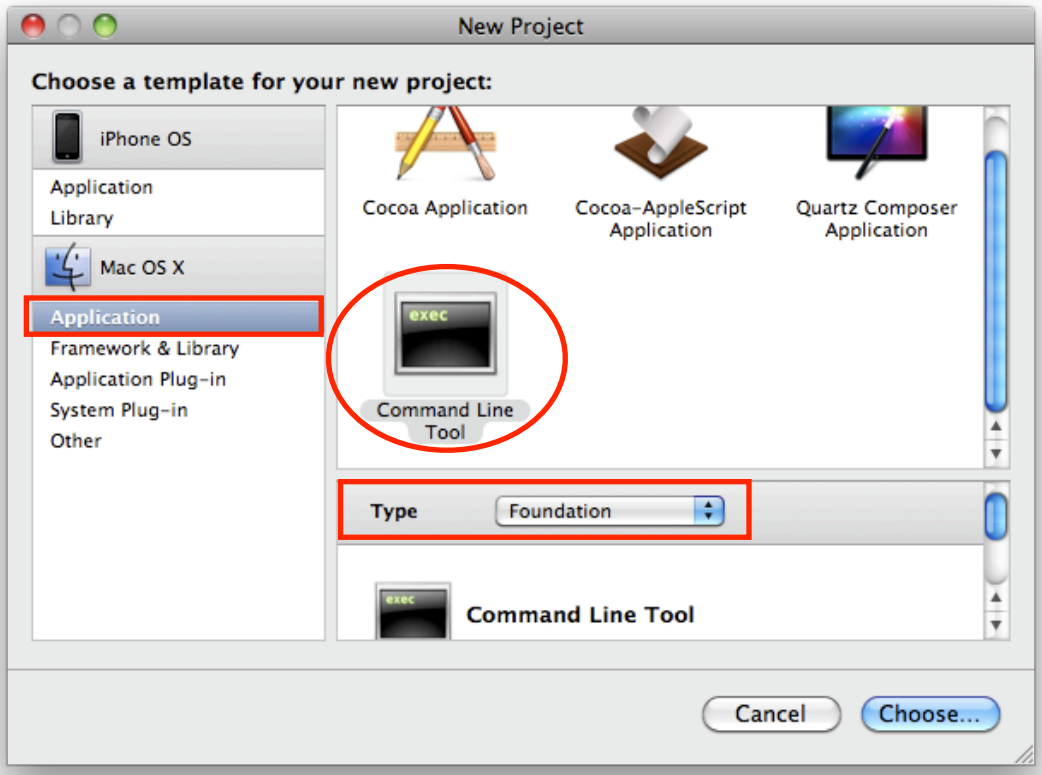
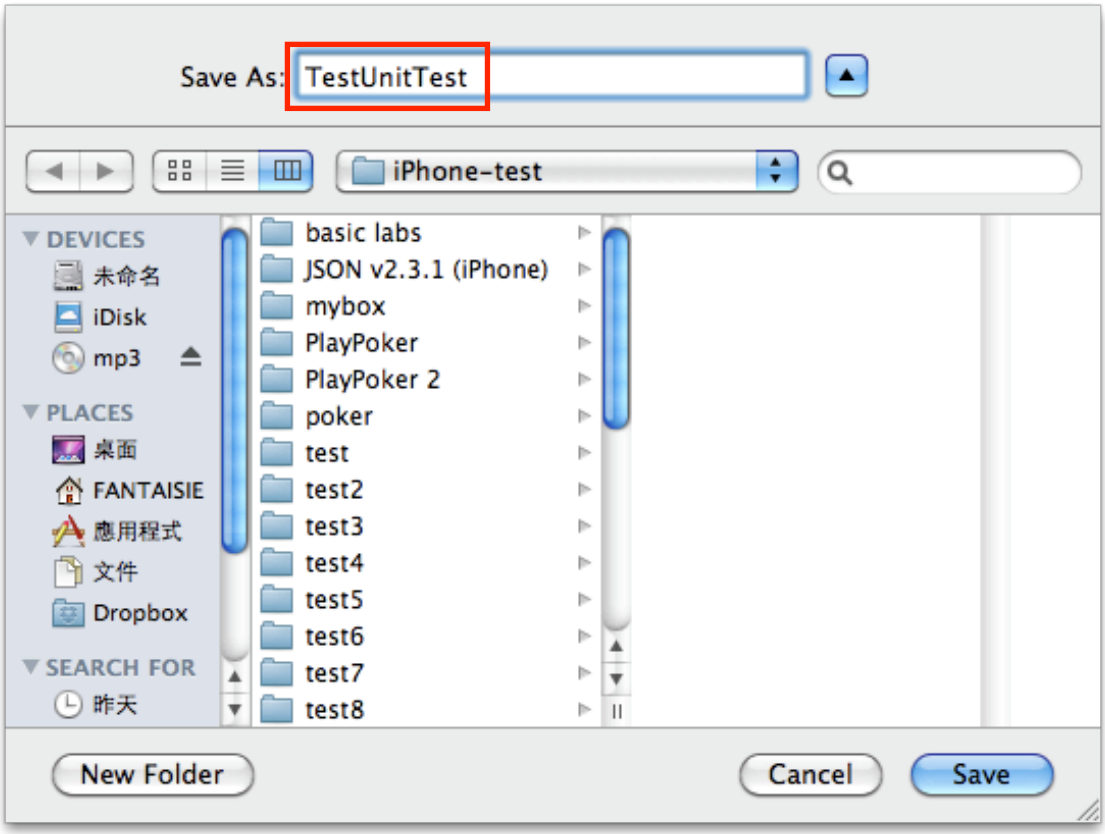
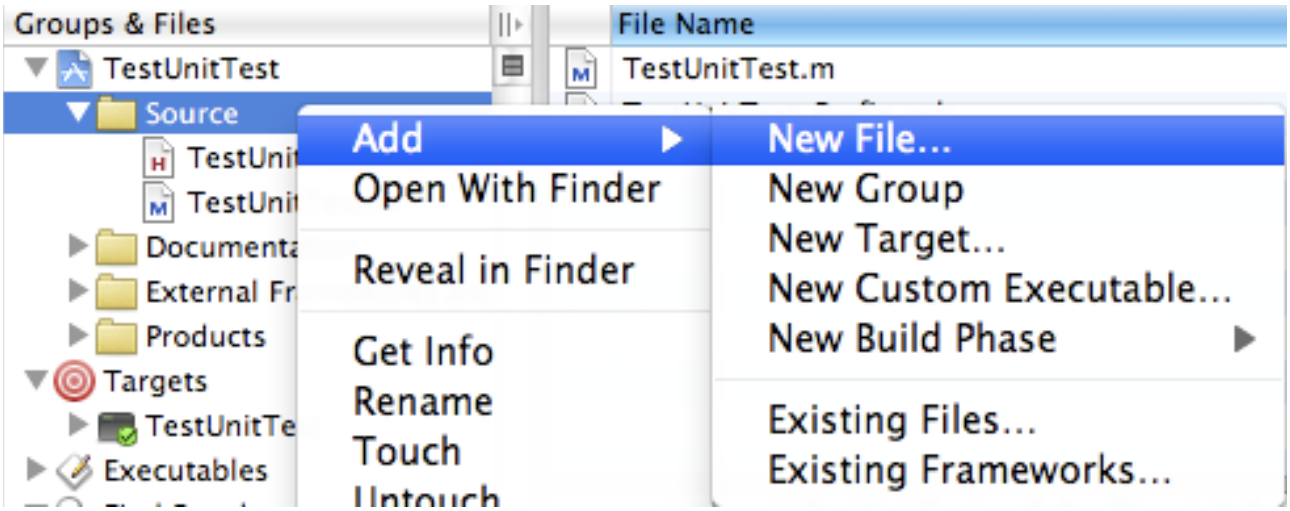
Lab TestUnitTest

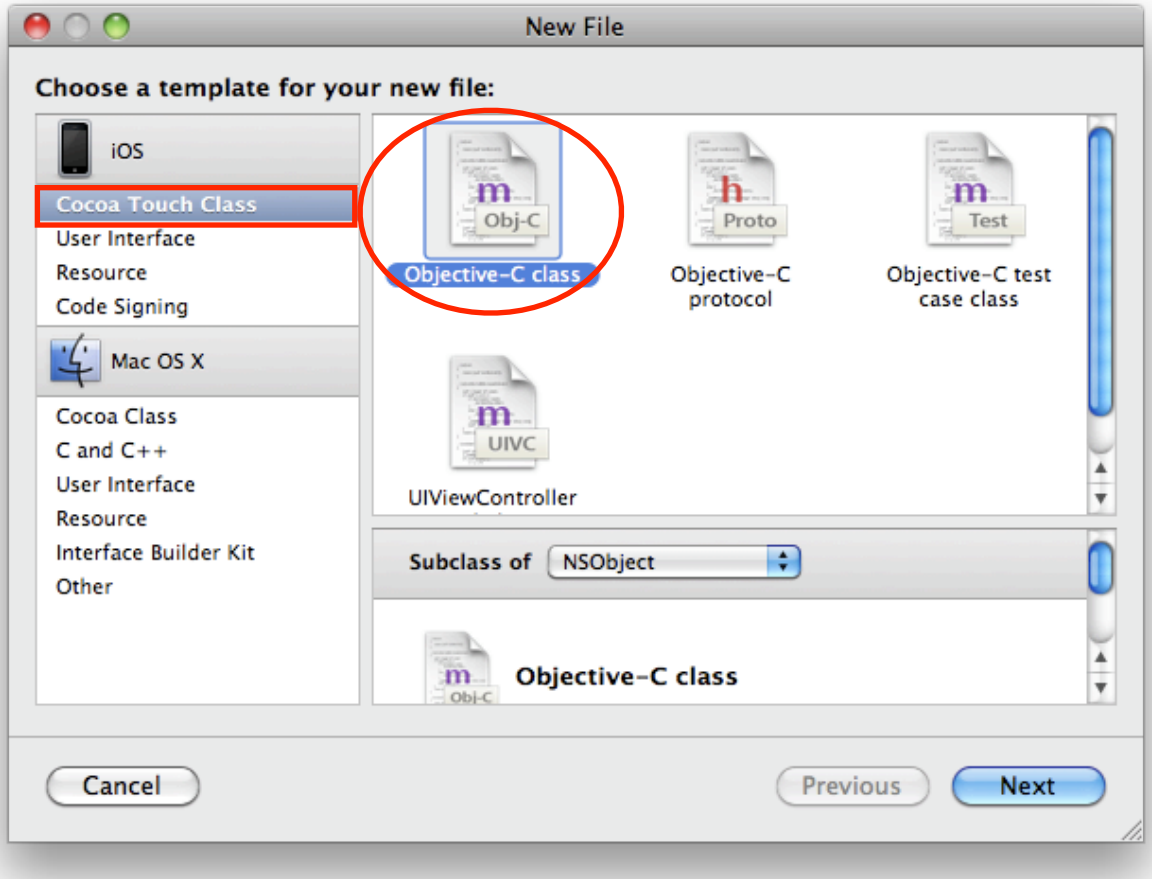
Step1. 在File開啓一個新的project, 選擇 MAC OS X的Command line Tool, Type選擇 Foundation, 將project命名為 TestUnitTest



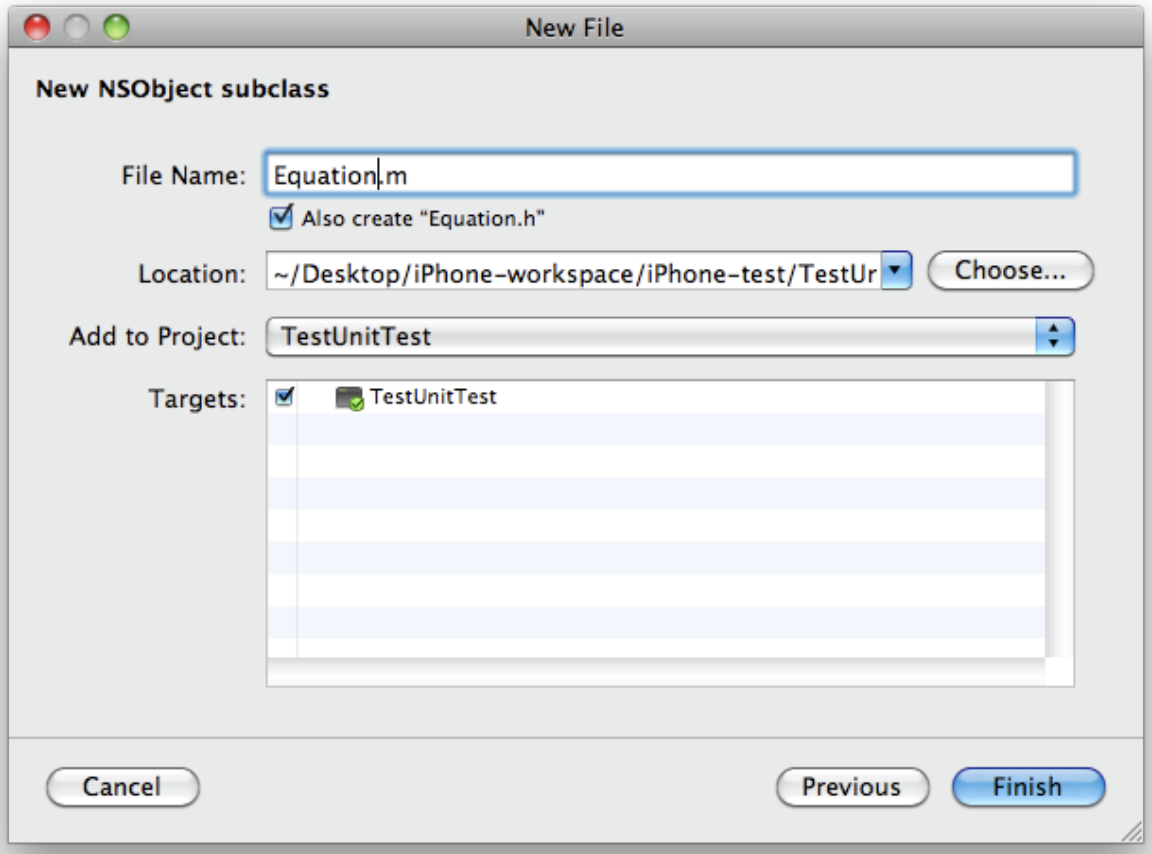


Step2. 在Xcode左邊Groups & Files 視窗中, 在Source這個路徑下點右鍵(若無滑鼠ctrl+點擊)選擇Add > New File...來增加新的Objective-C class





Step3. 我們要實作一個計算指數次方的程式, 將這個class命名為Equation.m



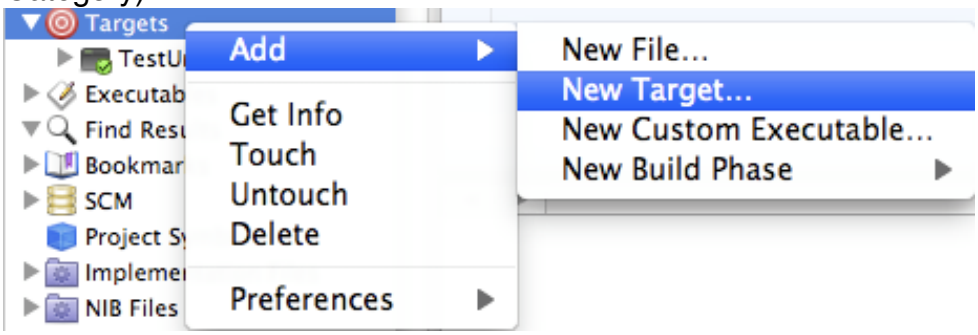
Step4. 在Equation.h裡面宣告我們要計算的基礎數字,對他做@property來自動以及可輸入參數值次方的指數運算

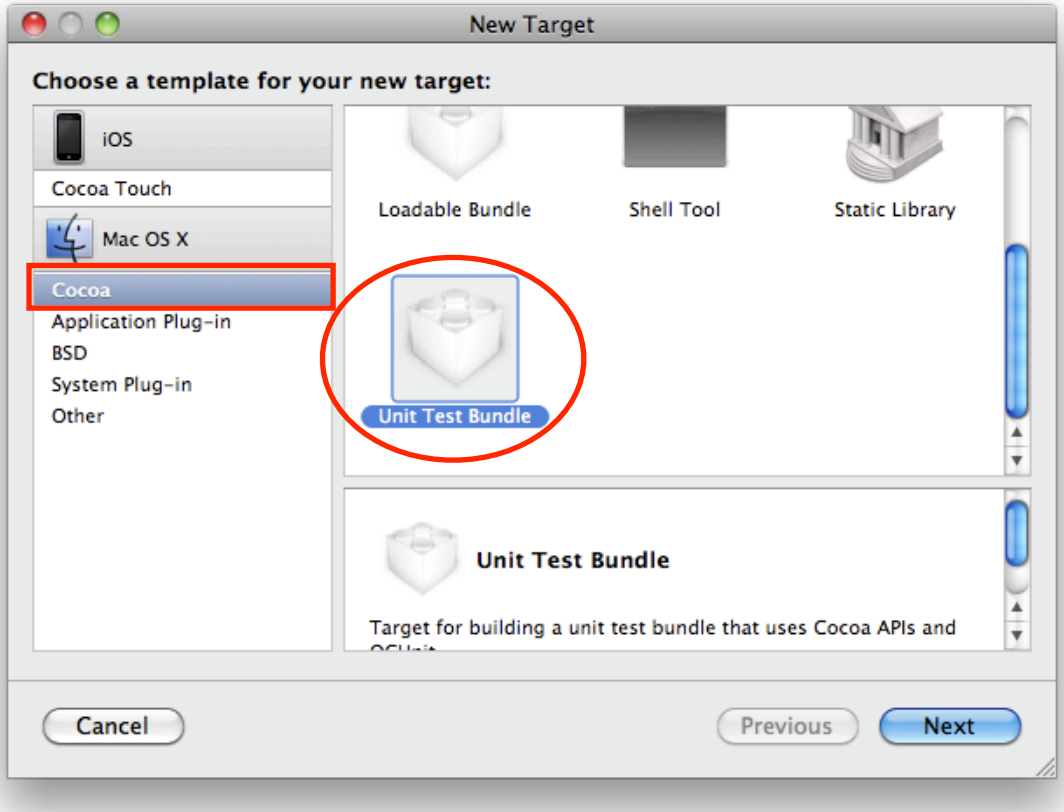
```
#import <Foundation/Foundation.h>


@interface Equation : NSObject {
    int number;
}
@property (assign) int number;

-(int) getResult:(int)exponent;

@end
```

Step5. 在Equation.m裡實作 `getResult:(int)exponent` 取得做輸入參數值次方的指數運算結果

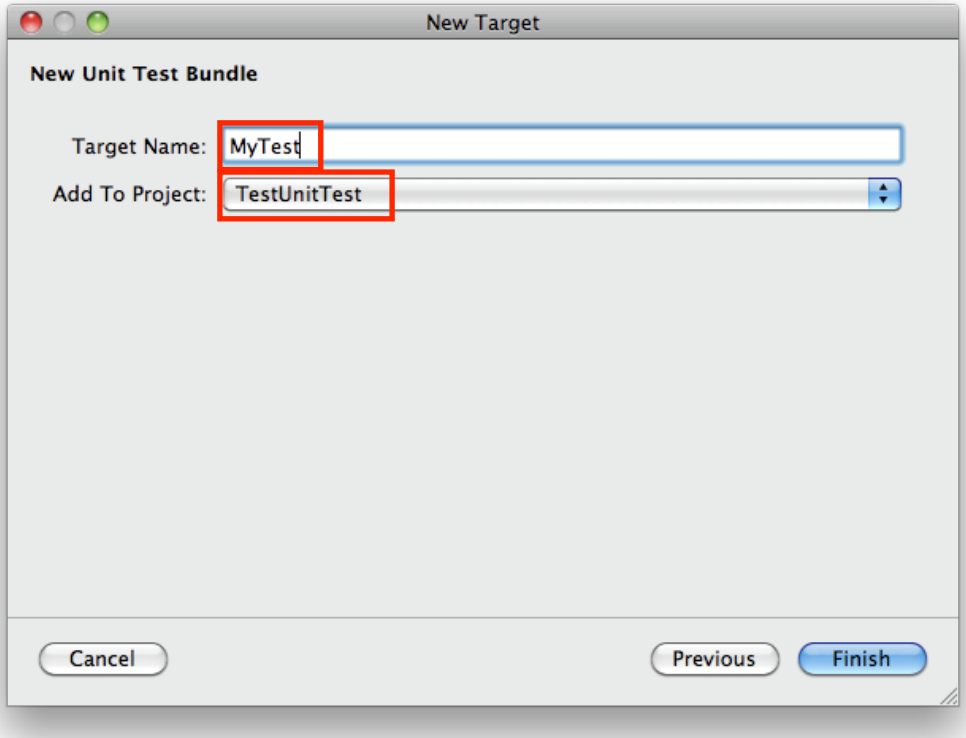
```
#import "Equation.h"

@implementation Equation
@synthesize number;

-(int) getResult:(int)exponent{
    return (number) ^ exponent;
}

@end
```

Step6. 接下來是來測試我們計算指數運算的method是否符合我們需要的邏輯,在Xcode左邊Groups & Files 視窗中, 在Targets這個路徑下點右鍵(若無滑鼠ctrl+點擊)選擇Add > New Target...來增加新的Test Target, 並選擇Unit Test Bundle (記得選擇Mac OS X的Cocoa Category)

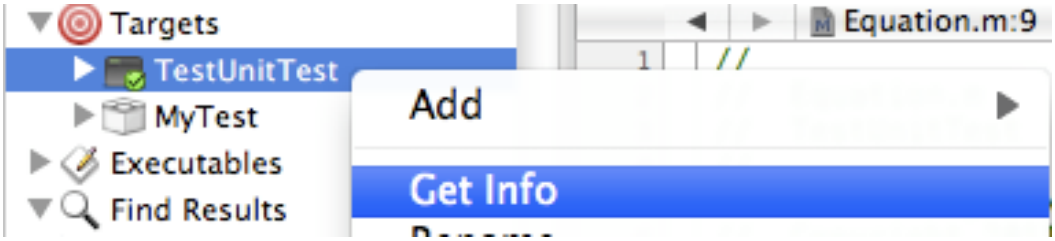




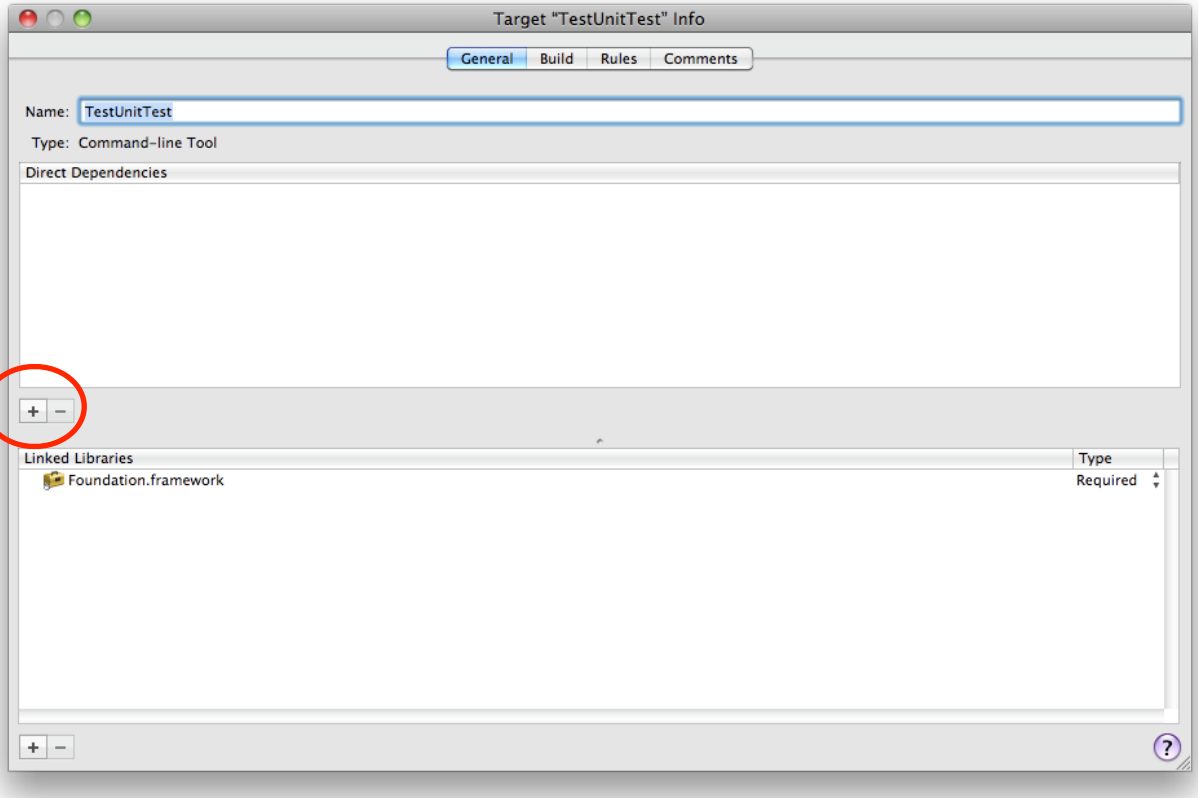
Step7. 將新增的New Unit Test Bundle 命名為MyTest



Step8. 接下來來Configure MyTest這個Test Target加到Project裡, 選擇Targets>TestUnitTest, 右鍵選擇Get Info

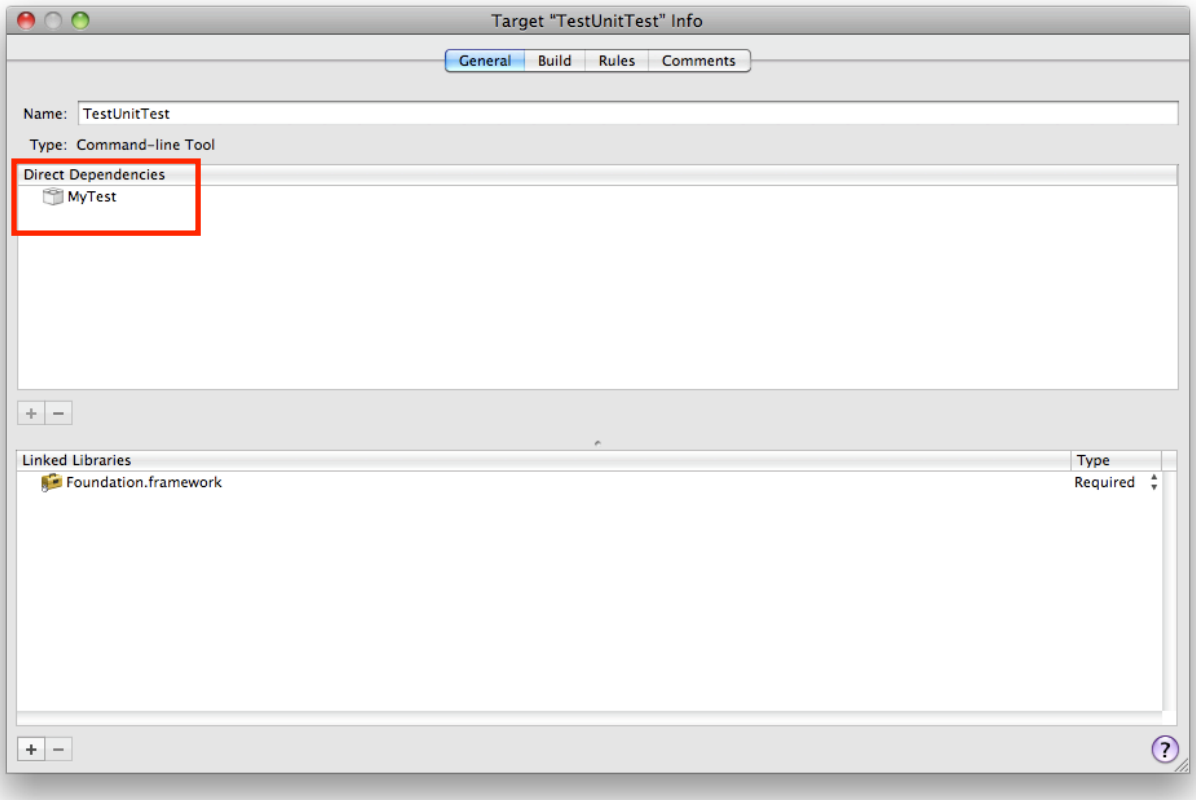


Step9. 點擊 Direct Dependencies 下的加號來加入我們MyTest這個Test Target

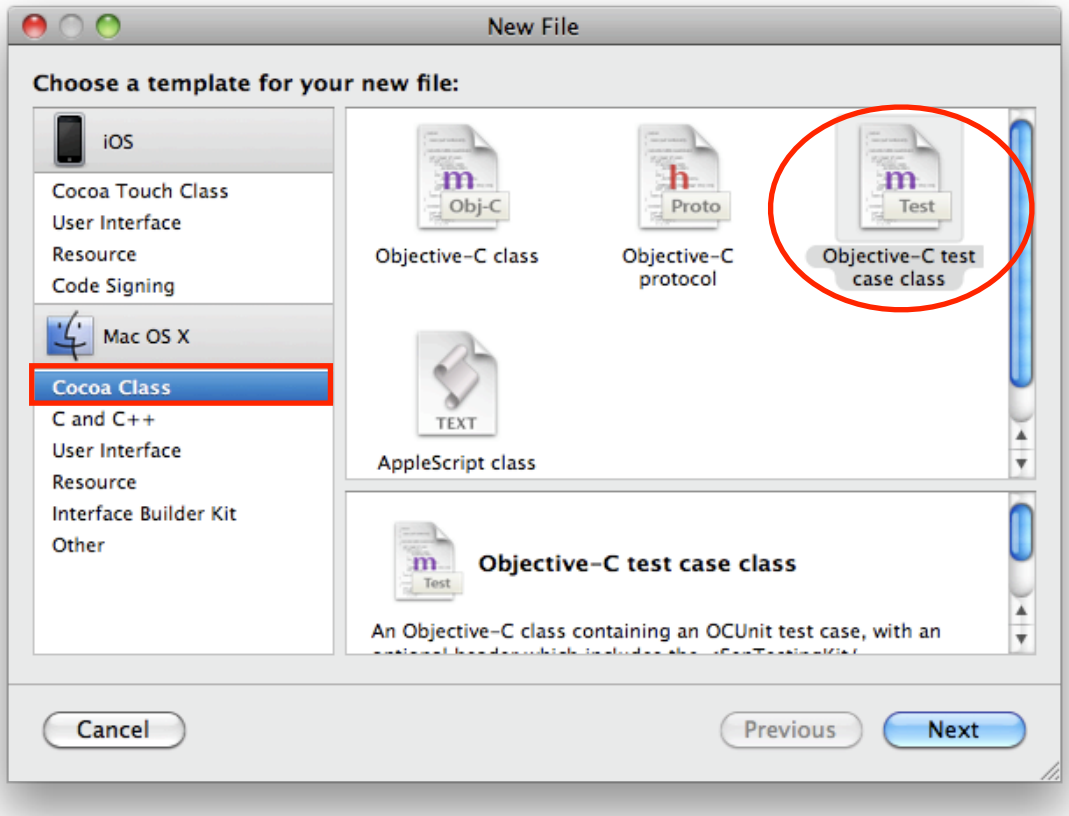


點選MyTest, 點Add Target

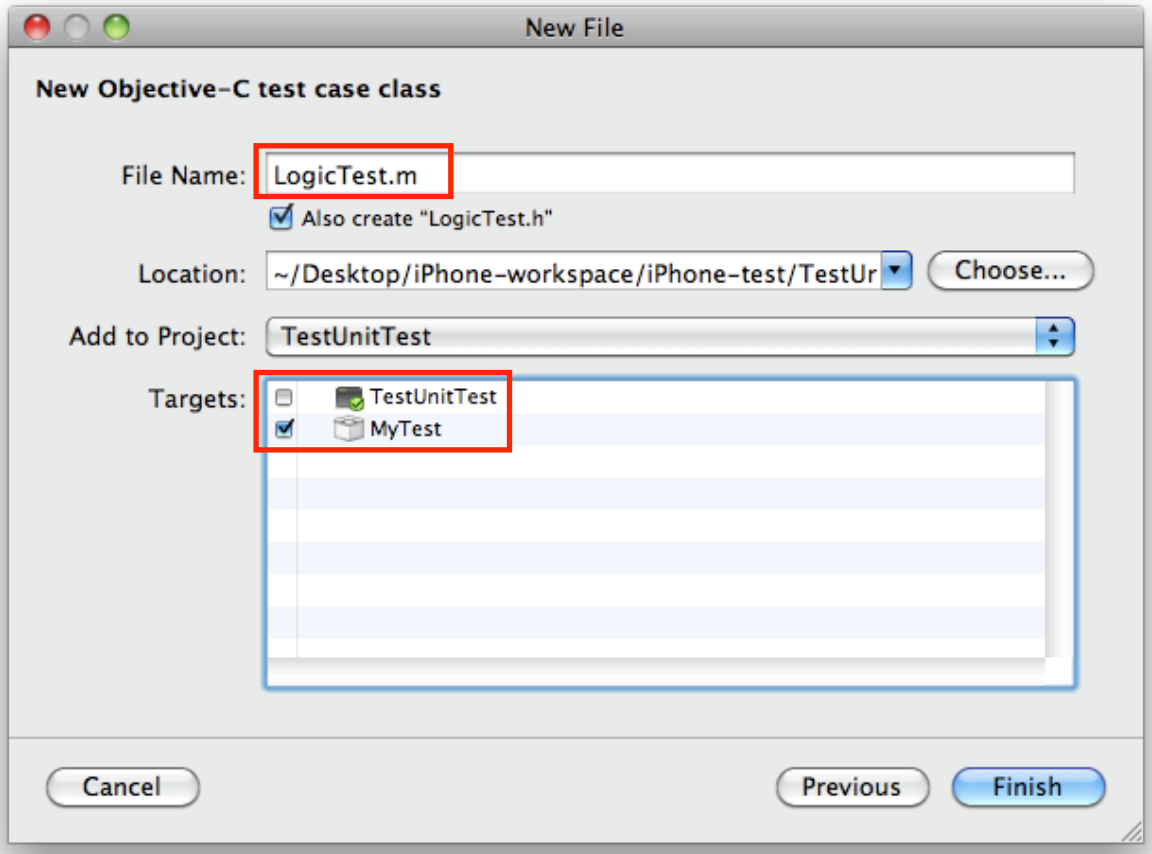
MyTest救出現在Direct Dependencies 欄中



Step10. 一樣在Xcode左邊Groups & Files 視窗中, 在Source這個路徑下點右鍵(若無滑鼠ctrl +點擊)選擇Add > New File...來增加新的Test Case (選擇Mac OS X Cocoa Category)



Step11. 命名為LogicTest, Target記得選取我們新增的Test Target “**MyTest**”



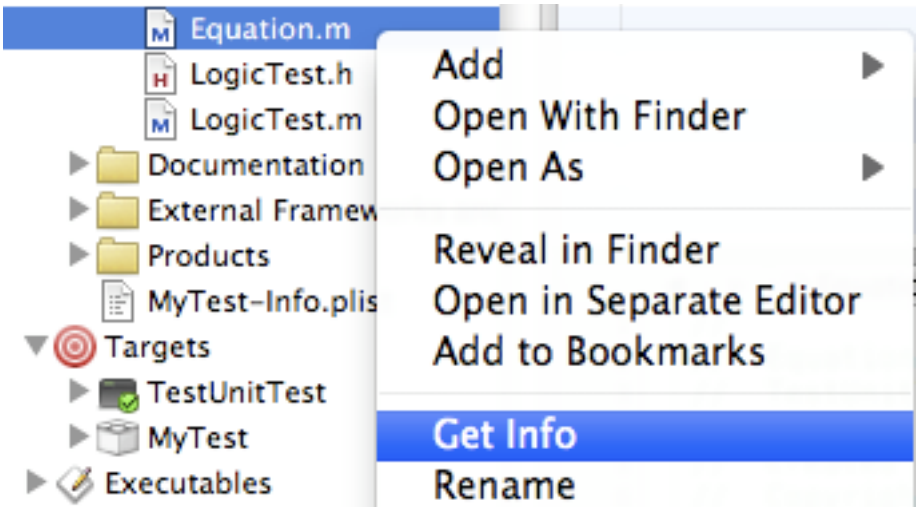
Step12. 在LogicTest.h裡,先 `#import "Equation.h"` ,再新增一個myEquation的物件

```
#import <SenTestingKit/SenTestingKit.h>
#import "Equation.h"

@interface LogicTest : SenTestCase {
    Equation *myEquation;
}

@end
```

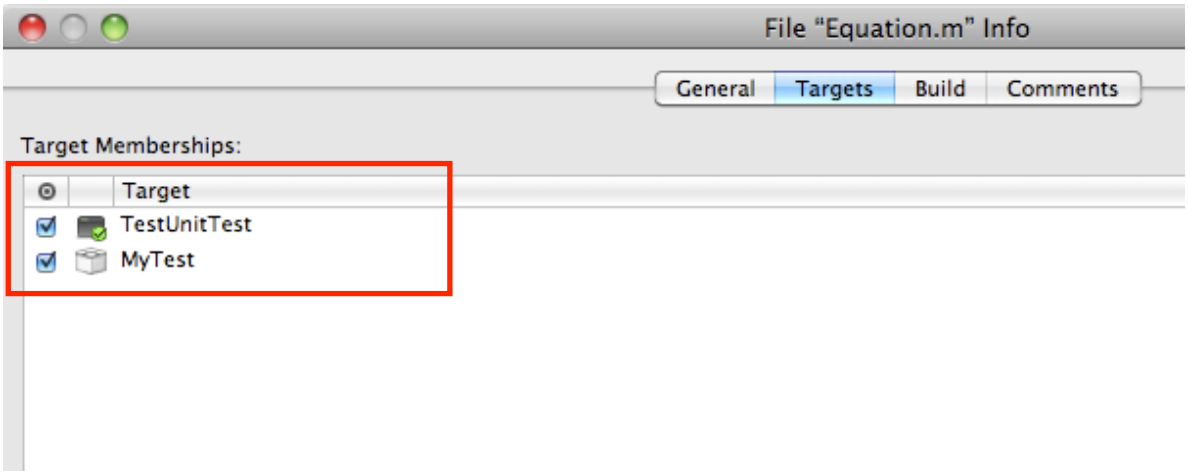
Step13. 在 LogicTest.m裡,實作我們的測試程式,先做SetUp的動作,在裡面把myEquation定址並實體化(new), 然後設定number的初始值為10, 並開始使用testSetter來設定測試條件, 我們預想10的三次方應該為1000, 若測試fail則印出後面的String, 其中含有目前計算指數運算的回傳值, 最後在tesrDown完成測試並將myEquation release,避免memory leak.

```
#import "LogicTest.h"


@implementation LogicTest

-(void) setUp {
 myEquation = [Equation new];
 [myEquation setNumber:10];
 NSLog(@"start point");
 }

 -(void) testSetter{
 STAssertEquals([myEquation getResult:3], 1000,
                [@"Should be 1000. Now it is " stringByAppendingString:
    [NSString stringWithFormat:@"%d", [myEquation getResult:3]]]);
 }

 -(void) tearDown {
 [myEquation release];
 NSLog(@"end point");
 }
@end
```

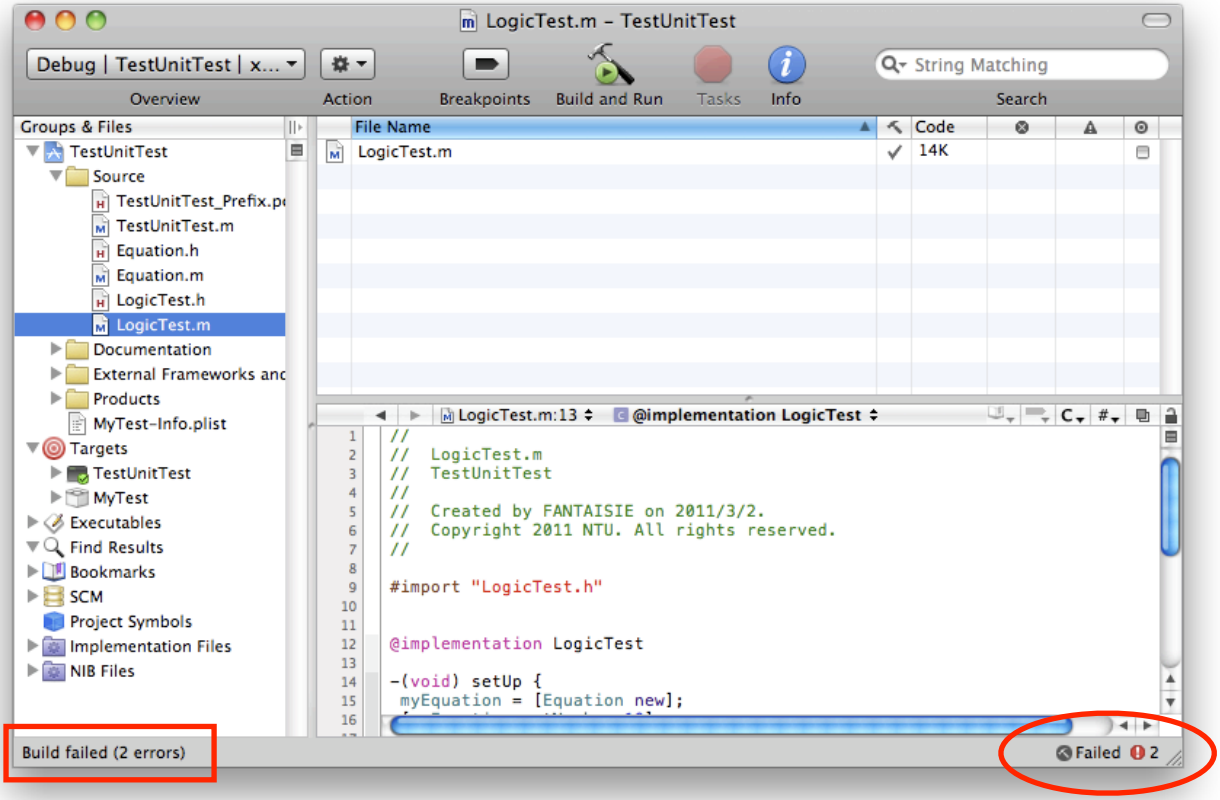
Step14. 在Equation.m上點右鍵選擇Get Info



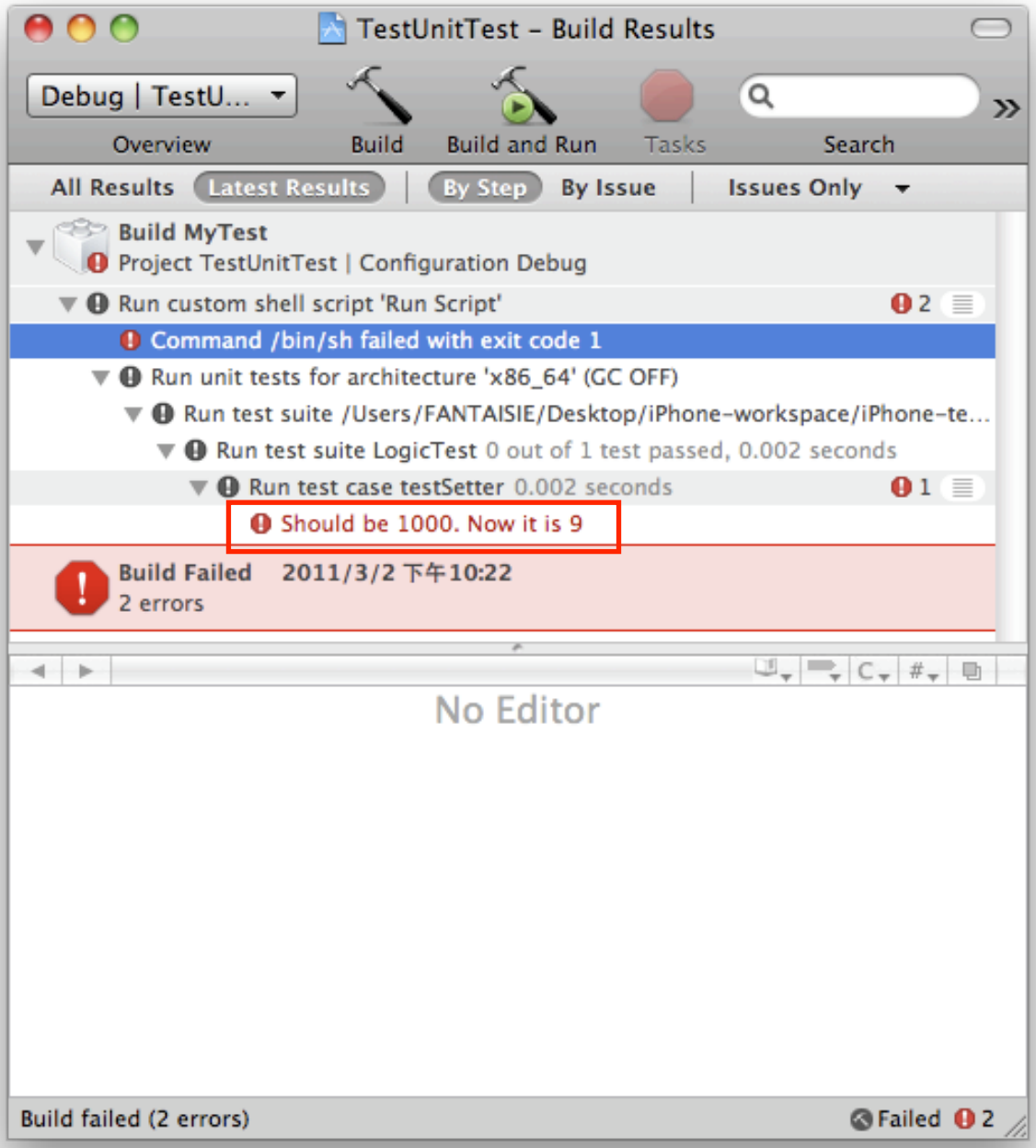
Step15. 在 Target Memberships 將兩個 Test Target都勾選



Step16. Build and Run (Command + enter)
在Xcode主頁上按下Build and Run, 或是在Build > Build and Run, 發現錯誤!



Step17. 發現錯誤! 預想值應為**1000**, 怎麼會目前的值是**9**?



Step18. 原來是在Equation.m裡面,我們的 `getResult:(int)exponent` 這個method回傳 `(number) ^ exponent` 其實是代表number和exponent兩個變數做XOR而不是number的exponent次方.

```
#import "Equation.h"

@implementation Equation
@synthesize number;

-(int) getResult:(int)exponent{
    return (number) ^ exponent;
}

@end
```

Step19. 修改 Equation.m裡的 `getResult:(int)exponent` 這個method的回傳值為 `pow((number), exponent)` 為C做指數運算的function, 語意即為我們想要的number的exponent次方.(註: 其實此回傳值為double,但在此不多加敘述研究)

```
#import "Equation.h"


@implementation Equation
@synthesize number;

-(int) getResult:(int)exponent{
    return pow((number), exponent);
}

@end
```

Step20. 再次Build and Run (Command + enter)
測試成功無Error, 意思就是計算指數運算的method符合我們想要的語意邏輯,接下來就可以繼續完成我們的application了!