```c
//
//  main.c
//  Array
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Demonstrates arrays.
//

#include <stdio.h>

int main(int argc, const char * argv[])
{
    // prompt user for number of exams
    int n;
    printf("Enter number of exams: ");
    scanf("%d", &n);

    // allocate memory for grades (on stack)
    int grades[n];

    // prompt user for exams' grades
    for (int i = 0; i < n; i++) {
        printf("Enter grade %d of %d: ", i+1, n);
        scanf("%d", &grades[i]);
    }

    // do something with grades...

    return 0;
}
```

```c
//
//  main.c
//  Conditions
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Reports whether user's input is positive, negative, or zero.
//

#include <stdio.h>

int main(int argc, const char * argv[])
{
    int n;
    printf("Enter an integer: ");
    scanf("%d", &n);
    if (n > 0) {
        printf("Thanks for the positive integer!\n");
    }
    else if (n < 0) {
        printf("Thanks for the negative integer!\n");
    }
    else {
        printf("Thanks for the zero!\n");
    }
    return 0;
}
```

```c
//
//  main.c
//  DoWhile
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Demonstrates a do-while loop.
//

#include <stdio.h>

int main(int argc, const char * argv[])
{
    int n;
    do {
        printf("Enter a positive integer: ");
        scanf("%d", &n);
    }
    while (n < 1);
    printf("Thanks for the positive integer!\n");
    return 0;
}
```

```c
//
//  main.c
//  Enum
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Demonstrates an enum (and a struct).
//

#include <stdio.h>

// gender
typedef enum {
    FEMALE,
    MALE
} genders;

// student
typedef struct {
    char *name;
    genders gender;
} student;

// prototype
void greet(student s);

int main(int argc, const char * argv[])
{
    // alice
    student alice;
    alice.name = "Alice";
    alice.gender = FEMALE;
    greet(alice);

    // bob
    student bob;
    bob.name = "Bob";
    bob.gender = MALE;
    greet(bob);

    return 0;
}

// greets student
void greet(student s)
{
    char *title = (s.gender == FEMALE) ? "Ms." : "Mr.";
    printf("Hello, %s %s.\n", title, s.name);
}
```

```c
//
//  main.c
//  For
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Demonstrates a for loop.
//

#include <stdio.h>

int main(int argc, const char * argv[])
{
    int n;
    printf("Enter a positive integer: ");
    scanf("%d", &n);
    for (int i = n; i > 0; i--) {
        printf("%d...\n", i);
    }
    printf("Blast off!\n");
    return 0;
}
```

```c
//
//  main.c
//  GetInt
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Gets an int from the user.
//

#include <stdio.h>

int main(int argc, const char * argv[])
{
    int n;
    printf("Enter an integer: ");
    scanf("%d", &n);
    printf("Thanks for the %d!\n", n);
    return 0;
}
```

```c
//
//  main.c
//  HelloC
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Says hello to the world in C.
//

#include <stdio.h>

int main (int argc, const char * argv[])
{
    int n = 50;
    printf("Hello, %s %d!\n", "World!", n);
    return 0;
}
```

```c
//
//  main.c
//  Malloc
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Demonstrates malloc.
//

#include <stdio.h>
#include <stdlib.h>

// prototype
int *get_grades(int exams);

int main (int argc, const char * argv[])
{
    // prompt user for number of exams
    int n;
    printf("Enter number of exams: ");
    scanf("%d", &n);

    // get grades
    int *grades = get_grades(n);

    // do something with grades...

    // free memory
    free(grades);

    return 0;
}

// gets grades
int *get_grades(int exams)
{
    // allocate memory for grades (on heap)
    int *grades = malloc(sizeof(int) * exams);

    // prompt user for exams' grades
    for (int i = 0; i < exams; i++) {
        printf("Enter grade %d of %d: ", i+1, exams);
        scanf("%d", &grades[i]);
    }
    return grades;
}
```

```c
//
//  main.c
//  Struct
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Demonstrates a struct.
//

#include <stdio.h>

// student
typedef struct {
    int age;
    char *name;
} student;

// prototype
void greet(student s);

int main (int argc, const char * argv[])
{
    // alice
    student alice;
    alice.age = 20;
    alice.name = "Alice";
    greet(alice);

    // bob
    student bob;
    bob.age = 21;
    bob.name = "Bob";
    greet(bob);

    return 0;
}

// greets student
void greet(student s)
{
    printf("Hello, %s.  I see that you are %d years old.\n", s.name, s.age);
}
```

```c
//
//  main.c
//  SwapFailure
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Fails to swap two variables' values.
//

#include <stdio.h>

// function prototype
void swap(int a, int b);


int main(int argc, const char * argv[])
{
    int x = 0;
    int y = 1;

    printf("x is %d\n", x);
    printf("y is %d\n", y);
    printf("Swapping x and y...\n");
    swap(x, y);
    printf("Success!\n");
    printf("x is %d\n", x);
    printf("y is %d\n", y);

    return 0;
}


//
// Swaps arguments' values.
//

void swap(int a, int b)
{
    int tmp = a;
    a = b;
    b = tmp;
}
```

```c
//
//  main.c
//  SwapSuccess
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Swaps two variables' values.
//

#include <stdio.h>

// function prototype
void swap(int *a, int *b);


int main(int argc, const char * argv[])
{
    int x = 0;
    int y = 1;

    printf("x is %d\n", x);
    printf("y is %d\n", y);
    printf("Swapping x and y...\n");
    swap(&x, &y);
    printf("Success!\n");
    printf("x is %d\n", x);
    printf("y is %d\n", y);

    return 0;
}


//
// Swaps arguments' values.
//

void swap(int *a, int *b)
{
    int tmp = *a;
    *a = *b;
    *b = tmp;
}
```

```c
//
//  main.c
//  While
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Demonstrates a while loop.
//

#include <stdio.h>

int main(int argc, const char * argv[])
{
    int n;
    printf("Enter a positive integer: ");
    scanf("%d", &n);
    while (n > 0) {
        printf("%d...\n", n);
        n--;
    }
    printf("Blast off!\n");
    return 0;
}
```

```objc
//
//  main.c
//  HelloObjC
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Says hello to the world in Objective-C.
//

#import <Foundation/Foundation.h>

int main(int argc, const char * argv[])
{
    @autoreleasepool {
        NSLog(@"Hello, World!");
    }
    return 0;
}
```

```
//
//  main.m
//  Students1
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Demonstrates use of a class.
//

#import <Foundation/Foundation.h>
#import "Student.h"

// prototype
void greet(Student *s);

int main(int argc, const char * argv[])
{
    @autoreleasepool {

        // Alice
        Student *alice = [Student alloc];
        alice->age = 20;
        alice->name = @"Alice";
        greet(alice);

        // Bob
        Student *bob = [Student alloc];
        bob->age = 21;
        bob->name = @"Bob";
        greet(bob);

    }
    return 0;
}

// greets student (via stderr)
void greet(Student *s)
{
    NSLog(@"Hello, %@.  I see that you are %d years old.\n", s->name, s->age);
}
```

```
//
//  Student.h
//  Students1
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Declares a student.
//

#import <Foundation/Foundation.h>

@interface Student : NSObject {
@public
    int age;
    NSString *name;
}
@end
```

```objc
//
//  Student.m
//  Students1
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Defines a student.
//

#import "Student.h"

@implementation Student
@end
```

```objc
//
//  main.m
//  Students2
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Demonstrates use of a class, getters, and setters.
//

#import <Foundation/Foundation.h>
#import "Student.h"

// prototype
void greet(Student *s);

int main(int argc, const char * argv[])
{
    @autoreleasepool {

        // Alice
        Student *alice = [[Student alloc] init];
        [[alice setAge:20] setName:@"Alice"];
        greet(alice);

        // Bob
        Student *bob = [[Student alloc] init];
        [bob setAge:21];
        [bob setName:@"Bob"];
        greet(bob);
    }
    return 0;
}

// greets student (via stderr)
void greet(Student *s)
{
    NSLog(@"Hello, %@.  I see that you are %d years old.\n", [s name], [s age]);
}
```

```objc
//
//  Student.h
//  Students2
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Declares a student with getters and setters.
//

#import <Foundation/Foundation.h>

@interface Student : NSObject {
    int _age;
    NSString *_name;
}

- (int)age;
- (Student *)setAge:(int)age;

- (NSString *)name;
- (Student *)setName:(NSString *)name;

@end
```

```
//
//  Student.m
//  Students2
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Defines a student with getters and setters.
//

#import "Student.h"

@implementation Student

- (int)age
{
    return _age;
}

- (Student *)setAge:(int)age
{
    _age = age;
    return self;
}

- (NSString *)name
{
    return _name;
}

- (Student *)setName:(NSString *)name
{
    _name = [name copy];
    return self;
}

@end
```

```objc
//
//  main.m
//  Students3
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Demonstrates use of properties.
//

#import <Foundation/Foundation.h>
#import "Student.h"

// prototype
void greet(Student *s);

int main(int argc, const char * argv[])
{
    @autoreleasepool {

        // Alice
        Student *alice = [[Student alloc] init];
        alice.age = 20;
        alice.name = @"Alice";
        greet(alice);

        // Bob
        Student *bob = [[Student alloc] init];
        bob.age = 21;
        bob.name = @"Bob";
        greet(bob);
    }
    return 0;
}

// greets student (via stderr)
void greet(Student *s)
{
    NSLog(@"Hello, %@.  I see that you are %d years old.\n", s.name, s.age);
}
```

```
//
//  Student.h
//  Students3
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Declares a student with properties.
//

#import <Foundation/Foundation.h>

@interface Student : NSObject {
    int _age;
    NSString *_name;
}

@property (assign, nonatomic, readwrite) int age;
@property (copy, nonatomic, readwrite) NSString *name;

- (int)age;
- (void)setAge:(int)age;

- (NSString *)name;
- (void)setName:(NSString *)name;

@end
```

```objc
//
//  Student.m
//  Students3
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Defines a student with properties.
//

#import "Student.h"

@implementation Student

- (int)age
{
    return _age;
}

- (void)setAge:(int)age
{
    _age = age;
}

- (NSString *)name
{
    return _name;
}

- (void)setName:(NSString *)name
{
    _name = [name copy];
}

@end
```

```
//
//  main.m
//  Students4
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Demonstrates use of synthesized properties.
//

#import <Foundation/Foundation.h>
#import "Student.h"

// prototype
void greet(Student *s);

int main(int argc, const char * argv[])
{
    @autoreleasepool {

        // Alice
        Student *alice = [[Student alloc] init];
        alice.age = 20;
        alice.name = @"Alice";
        greet(alice);

        // Bob
        Student *bob = [[Student alloc] init];
        bob.age = 21;
        bob.name = @"Bob";
        greet(bob);
    }
    return 0;
}

// greets student (via stderr)
void greet(Student *s)
{
    NSLog(@"Hello, %@.  I see that you are %d years old.\n", s.name, s.age);
}
```

```objc
//
//  Student.h
//  Students4
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Declares a student with synthesized properties.
//

#import <Foundation/Foundation.h>

@interface Student : NSObject {
}

@property (assign, nonatomic, readwrite) int age;
@property (copy, nonatomic, readwrite) NSString *name;

@end
```

```
//
//  Student.m
//  Students4
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Defines a student with synthesized properties.
//

#import "Student.h"

@implementation Student
@end
```

```
//
//  main.m
//  Students5
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Demonstrates use of (mostly) synthesized properties.
//

#import <Foundation/Foundation.h>
#import "Student.h"

// prototype
void greet(Student *s);

int main(int argc, const char * argv[])
{
    @autoreleasepool {

        // Alice
        Student *alice = [[Student alloc] init];
        alice.age = 20;
        alice.name = @"Alice";
        greet(alice);

        // Bob
        Student *bob = [[Student alloc] init];
        bob.age = 21;
        bob.name = @"Bob";
        greet(bob);

        // David
        Student *david = [[Student alloc] init];
        david.age = 29;
        david.name = @"David";
        greet(david);
    }
    return 0;
}

// greets student (via stderr)
void greet(Student *s)
{
    NSLog(@"Hello, %@.  I see that you are %d years old.\n", s.name, s.age);
}
```

```
//
//  Student.h
//  Students5
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Declares a student with (mostly) synthesized properties.
//

#import <Foundation/Foundation.h>

@interface Student : NSObject {
}

@property (assign, nonatomic, readwrite) int age;
@property (copy, nonatomic, readwrite) NSString *name;

@end
```

```objc
//
//  Student.m
//  Students5
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Defines a student with (mostly) synthesized properties.
//

#import "Student.h"

@implementation Student

- (void)setName:(NSString *)name
{
    if ([name isEqualToString:@"David"]) {
        _name = @"Dummy";
    }
    else {
        _name = [name copy];
    }
}


@end
```

```objc
//
//  main.m
//  Students6
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Demonstrates an init... method.
//

#import <Foundation/Foundation.h>
#import "Student.h"

// prototype
void greet(Student *s);

int main(int argc, const char * argv[])
{
    @autoreleasepool {

        // Alice
        Student *alice = [[Student alloc] initWithName:@"Alice" andAge:20];
        greet(alice);

        // Bob
        Student *bob = [[Student alloc] initWithName:@"Bob" andAge:21];
        greet(bob);

        // John
        Student *john = [[Student alloc] init];
        greet(john);
    }
    return 0;
}

// greets student (via stderr)
void greet(Student *s)
{
    NSLog(@"Hello, %@.  I see that you are %d years old.\n", s.name, s.age);
}
```

```objc
//
//  Student.h
//  Students6
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Declares a student with an init... method.
//

#import <Foundation/Foundation.h>

@interface Student : NSObject {
}

@property (assign, nonatomic, readwrite) int age;
@property (copy, nonatomic, readwrite) NSString *name;

- (id)initWithName:(NSString *)name andAge:(int)age;

@end
```

```objc
//
//  Student.m
//  Students6
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Defines a student with an init... method.
//

#import "Student.h"

@implementation Student

- (id)init
{
    self = [self initWithName:@"John" andAge:406];
    return self;
}

- (id)initWithName:(NSString *)name andAge:(int)age
{
    if (self = [super init])
    {
        self.age = age;
        self.name = name;
    }
    return self;
}

@end
```

```objc
//
//  main.m
//  Students7
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Demonstrates mutable arrays.
//

#import <Foundation/Foundation.h>
#import "Student.h"

// prototype
void greet(Student *s);

int main(int argc, const char * argv[])
{
    @autoreleasepool {

        // allocate array for students
        NSMutableArray *students = [[NSMutableArray alloc] init];

        // Alice
        Student *alice = [[Student alloc] initWithName:@"Alice" andAge:20];
        [students addObject:alice];

        // Bob
        [students addObject:[[Student alloc] initWithName:@"Bob" andAge:21]];

        // greet then release each student
        for (Student *s in students) {
            greet(s);
        }
    }
    return 0;
}

// greets student (via stderr)
void greet(Student *s)
{
    NSLog(@"Hello, %@.  I see that you are %d years old.\n", s.name, s.age);
}
```

```
//
//  Student.h
//  Students7
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Declares a student with init methods.
//

#import <Foundation/Foundation.h>

@interface Student : NSObject {
}

@property (assign, nonatomic, readwrite) int age;
@property (copy, nonatomic, readwrite) NSString *name;

- (id)initWithName:(NSString *)name andAge:(int)age;

@end
```

```
//
//  Student.m
//  Students7
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Defines a student with init methods.
//

#import "Student.h"

@implementation Student

- (id)init
{
    self = [self initWithName:@"John" andAge:406];
    return self;
}

- (id)initWithName:(NSString *)name andAge:(int)age
{
    if (self = [super init])
    {
        self.age = age;
        self.name = name;
    }
    return self;
}

@end
```

```objc
//
//  main.m
//  Students8
//
//  Robert Bowden
//  Harvard University
//  rob@cs.harvard.edu
//
//  Demonstrates sorting an array using a selector
//

#import <Foundation/Foundation.h>
#import "Student.h"

// prototype
void greet(Student *s);

int main(int argc, const char * argv[])
{
    @autoreleasepool {

        // allocate array for students
        NSMutableArray *students = [[NSMutableArray alloc] init];

        // R.J.
        [students addObject:[[Student alloc] initWithName:@"R.J." andAge:21]];

        // Chris
        [students addObject:[[Student alloc] initWithName:@"Chris" andAge:18]];

        // Rob
        [students addObject:[[Student alloc] initWithName:@"Rob" andAge:22]];

        // David
        [students addObject:[[Student alloc] initWithName:@"David" andAge:85]];

        // sort the array
        NSArray *sortedStudents = [students sortedArrayUsingSelector:@selector(compare:)];

        // greet then release each student
        for (Student *s in sortedStudents) {
            greet(s);
        }
    }
    return 0;
}

// greets student (via stderr)
```

```
void greet(Student *s)
{
    NSLog(@"Hello, %@.  I see that you are %d years old.\n", s.name, s.age);
}
```

```objc
//
//  Student.h
//  Students8
//
//  Robert Bowden
//  Harvard University
//  rob@cs.harvard.edu
//
//  Declares a student with init methods.
//

#import <Foundation/Foundation.h>

@interface Student : NSObject {
}

@property (assign, nonatomic, readwrite) int age;
@property (copy, nonatomic, readwrite) NSString *name;

- (id)initWithName:(NSString *)name andAge:(int)age;
- (NSComparisonResult)compare:(Student *)otherStudent;


@end
```

```
//
//  Student.m
//  Students8
//
//  Robert Bowden
//  Harvard University
//  rob@cs.harvard.edu
//
//  Defines a student with init methods.
//

#import "Student.h"

@implementation Student

- (id)init
{
    self = [self initWithName:@"John" andAge:406];
    return self;
}

- (id)initWithName:(NSString *)name andAge:(int)age
{
    if (self = [super init])
    {
        self.age = age;
        self.name = name;
    }
    return self;
}

- (NSComparisonResult)compare:(Student *)otherStudent {
    return [self.name compare:otherStudent.name];
}

@end
```

```objc
//
//  main.m
//  Students9
//
//  Robert Bowden
//  Harvard University
//  rob@cs.harvard.edu
//
//  Demonstrates sorting an array using a selector from a category
//

#import <Foundation/Foundation.h>
#import "Student+StudentCompare.h"

// prototype
void greet(Student *s);

int main(int argc, const char * argv[])
{
    @autoreleasepool {

        // allocate array for students
        NSMutableArray *students = [[NSMutableArray alloc] init];

        // R.J.
        [students addObject:[[Student alloc] initWithName:@"R.J." andAge:21]];

        // Chris
        [students addObject:[[Student alloc] initWithName:@"Chris" andAge:18]];

        // Rob
        [students addObject:[[Student alloc] initWithName:@"Rob" andAge:22]];

        // David
        [students addObject:[[Student alloc] initWithName:@"David" andAge:85]];

        // sort the array
        NSArray *sortedStudents = [students sortedArrayUsingSelector:@selector(compare:)];

        // greet then release each student
        for (Student *s in sortedStudents) {
            greet(s);
        }
    }
    return 0;
}

// greets student (via stderr)
```

```
void greet(Student *s)
{
    NSLog(@"Hello, %@.  I see that you are %d years old.\n", s.name, s.age);
}
```

```objc
//
//  Student+StudentCompare.h
//  Students9
//
//  Robert Bowden
//  Harvard University
//  rob@cs.harvard.edu
//
//  Demonstrates sorting an array using a selector from a category
//

#import "Student.h"

@interface Student (StudentCompare)

- (NSComparisonResult)compare:(Student *)otherStudent;

@end
```

```
//
//  Student+StudentCompare.m
//  Students9
//
//  Robert Bowden
//  Harvard University
//  rob@cs.harvard.edu
//
//  Demonstrates sorting an array using a selector from a category
//

#import "Student+StudentCompare.h"

@implementation Student (StudentCompare)

- (NSComparisonResult)compare:(Student *)otherStudent {
    return [self.name compare:otherStudent.name];
}

@end
```

```
//
//  Student.h
//  Students9
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Declares a student with init methods.
//

#import <Foundation/Foundation.h>

@interface Student : NSObject {
}

@property (assign, nonatomic, readwrite) int age;
@property (copy, nonatomic, readwrite) NSString *name;

- (id)initWithName:(NSString *)name andAge:(int)age;

@end
```

```
//
//  Student.m
//  Students9
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Defines a student with init methods.
//

#import "Student.h"

@implementation Student

- (id)init
{
    self = [self initWithName:@"John" andAge:406];
    return self;
}

- (id)initWithName:(NSString *)name andAge:(int)age
{
    if (self = [super init])
    {
        self.age = age;
        self.name = name;
    }
    return self;
}

@end
```

```
//
//  main.m
//  Students10
//
//  Robert Bowden
//  Harvard University
//  rob@cs.harvard.edu
//
//  Demonstrates sorting an array using a block
//

#import <Foundation/Foundation.h>
#import "Student.h"

// prototype
void greet(Student *s);

int main(int argc, const char * argv[])
{
    @autoreleasepool {

        // allocate array for students
        NSMutableArray *students = [[NSMutableArray alloc] init];

        // R.J.
        [students addObject:[[Student alloc] initWithName:@"R.J." andAge:20]];

        // Chris
        [students addObject:[[Student alloc] initWithName:@"Chris" andAge:20]];

        // Rob
        [students addObject:[[Student alloc] initWithName:@"Rob" andAge:20]];

        // David
        [students addObject:[[Student alloc] initWithName:@"David" andAge:21]];

        // store the block in a variable called "func". Alternatively, could
        // inline the block in the line below, similar to anonymous functions
        // in JavaScript
        NSComparisonResult (^func)(Student *, Student *) = ^NSComparisonResult(Student *a, Student *b) {
            return [a.name compare:b.name];
        };

        // sort the array using our block
        NSArray *sortedStudents = [students sortedArrayUsingComparator:func];

        // greet then release each student
        for (Student *s in sortedStudents) {
            greet(s);
        }
    }
    return 0;
}

// greets student (via stderr)
void greet(Student *s)
{
    NSLog(@"Hello, %@.  I see that you are %d years old.\n", s.name, s.age);
}
```

```
//
//  Student.h
//  Students10
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Declares a student with init methods.
//

#import <Foundation/Foundation.h>

@interface Student : NSObject {
}

@property (assign, nonatomic, readwrite) int age;
@property (copy, nonatomic, readwrite) NSString *name;

- (id)initWithName:(NSString *)name andAge:(int)age;

@end
```

```
//
//  Student.m
//  Students10
//
//  David J. Malan
//  Harvard University
//  malan@harvard.edu
//
//  Defines a student with init methods.
//

#import "Student.h"

@implementation Student

- (id)init
{
    self = [self initWithName:@"John" andAge:406];
    return self;
}

- (id)initWithName:(NSString *)name andAge:(int)age
{
    if (self = [super init])
    {
        self.age = age;
        self.name = name;
    }
    return self;
}

@end
```