# UNIVERSIDAD DEL MAR

**Campus Puerto Escondido**

**TECNOLOGÍAS DE INFORMACIÓN II**

**Programación móvil: IOS**

**Profesor:** M. en C. Carlos Rojas Sánchez

**Alumno:** Emanuel Ruiz Triste

**Licenciatura en Informática**

Puerto Escondido, Oax. 14 de marzo de 2016

## Primer Nivel

```
NSLog(@"Hello, Emanuel");  //Salida de texto a consola
NSString *firstName = @"Emanuel";  //Declaración de una variable string
NSLog(firstName);  //Salida a consola de la variable firstName
NSLog(@"Hello there, %@.", firstName);  //Salida con formato de la variable antes declarada
NSLog(@"%@ %@", firstName, firstName);  //salida con formato y concatenación de variable firstName
NSString *lastName = @"Ruiz";   //Declaración de otra variable string
NSLog(@"%@ %@", firstName, lastName);   //Salida con formateo y concatenación de variables String
NSNumber *age = @23;   //Declaración de variable numero o entero
NSLog(@"%@ is %@ years old", firstName, age);   //Salida de string concatenado
NSArray *apps = @[@"AngryFowl", @"Lettertouch", @"Tweetrobot"];   //Declaración de un arreglo
NSLog(@"%@", apps[1]);   //Visualización del elemento 2 del arreglo apps
apps = @[@"AngryFowl", @"Lettertouch", @"Tweetrobot", @"Instacanvas"];   //Agregar otro elemento al arreglo
NSDictionary *appRatings = @{@"AngryFowl": @3, @"Lettertouch": @5};   //Declaración de diccionarios
NSLog(@"Lettertouch has a rating of %@.", appRatings[@"Lettertouch"]);   //Salida de un valor del diccionario
```

## Segundo Nivel

```
[tryobjc completeThisChallenge];   //Envío de mensaje a un objeto

//Declaración y salida de un arreglo
NSArray *foods = @[@"tacos", @"burgers"];
NSLog(@"%@", foods);

//Declaración de un array y un string que almacena la descripción de dicho array
NSArray *foods = @[@"tacos", @"burgers"];
NSString *result = [foods description];
NSLog(@"%@", result);

NSString *city = @"Ice World";   //Declaración de un string
NSUInteger cityLength = [city length];   //Declaración de variable entera, almacena el tamaño del string
NSLog(@"City has %lu characters", cityLength);  //Salida de un entero con formato

NSNumber *higgiesAge = @6;   //Variable número
NSNumber *phoneLives = @3;   //Variable número
NSUInteger higgiesAgeInt = [higgiesAge unsignedIntegerValue];  //Conversión de número a entero
NSUInteger phoneLivesInt = [phoneLives unsignedIntegerValue];  //Conversión de número a entero
NSUInteger higgiesRealAge = higgiesAgeInt * phoneLivesInt;  //Multiplicación de enteros
NSLog(@"Higgie is actually %lu years old.", higgiesRealAge);  //Salida de entero

NSString *firstName = @"Emanuel";  //Variable string
NSString *lastName = @"Ruiz";  //Variable string
NSString *fullName = [firstName stringByAppendingString:lastName];  //Concatena Strings
NSLog(@"%@", fullName);   //Salida de string

NSString *firstName = @"Emanuel";  //Variable de string
NSString *lastName = @"Ruiz";  //Variable de string
//Concatena variables string con un espacio
NSString *fullName = [[firstName stringByAppendingString:@" "] stringByAppendingString:lastName];
NSLog(@"%@", fullName);  //Salida de variable string
```

```

NSString *firstName = @"Emanuel";  //Variable de string
NSString *lastName = @"Ruiz";  //Variable de string
NSString *fullName = [[firstName stringByAppendingString:@" "] stringByAppendingString:lastName];
//Remplaza variables string
NSString *replaced = [fullName stringByReplacingOccurrencesOfString:firstName withString:lastName];
NSLog(@"%@", replaced);

NSString *firstName = @"Emanuel";  //Variable de string
NSString *copy = [NSString stringWithString:firstName];  //Hace una copia de la variable string
NSLog(@"%@ is a copy of %@", copy, firstName);  //Muestra el valor de la copia

NSString *firstName = @"Emanuel";  //Variable de string
NSString *copy = [[NSString alloc] initWithString:firstName];  //Hace una copia de la variable string
NSLog(@"%@ is a copy of %@", copy, firstName);  //Salida de un string

NSString *firstName = @"Emanuel";  //Variable de string
NSString *lastName = @"Ruiz";  //Variable de string
NSString *fullName = [NSString stringWithFormat:@"%@ %@", firstName, lastName];  //Concatena con formato
NSLog(@"%@", fullName);  //salida de la variable
```

## Tercer Nivel

```
//Bloque para un condicional, if
BOOL mrHiggieIsMean = YES
if (mrHiggieIsMean) {
  NSLog(@"Confirmed: he is super mean");
}

//Bloque para condicional if-else
BOOL mrHiggieIsMean = [mrHiggie areYouMean];
if (mrHiggieIsMean) {
  NSLog(@"Confirmed: he is super mean");
} else {
  NSLog(@"No, actually he's really nice");
}

//Bloque de código para if anidado para número
NSNumber *meannessScale = [mrHiggie meannessScale];
if([meannessScale intValue] < 4) {
  NSLog(@"Mr. Higgie is on the nice side");
} else if([meannessScale intValue] < 8) {
  NSLog(@"Mr. Higgie is sorta nice but not really");
} else {
  NSLog(@"Mr. Higgie is definitely mean");
}

// Bloque de código para if anidado para string
NSString *hat = [mrHiggie currentHat];
if([hat isEqualToString:@"Sombrero"]) {
  NSLog(@"Ese es un muy buen sombrero");
```

```
    }
        case DayOfWeekSaturday:
    case DayOfWeekSunday: {
        [mrHiggie setCurrentHat:@"AstronautHelmet"];
        break;
    }
}
NSLog(@"Mr. Higgie is wearing: %@", [mrHiggie currentHat]);

//Ciclo for, tiene una variable string llamada hat. Esto usando un arreglo
e imprime una frase según el valor
NSArray *newHats = @[@"Cowboy", @"Conductor", @"Baseball"];
for (NSString *hat in newHats) {
  NSLog(@"Trying on new %@ hat", hat);
  if([mrHiggie tryOnHat:hat]) {
    NSLog(@"Mr. Higgie loves it");
  } else {
    NSLog(@"Mr. Higgie hates it");
  }
}

//Enumeración como diccionario, no es limitado
NSDictionary *funnyWords = @{
  @"Schadenfreude": @"pleasure derived by someone from another person's
misfortune.",
  @"Portmanteau": @"consisting of or combining two or more separable
aspects or qualities",
  @"Penultimate": @"second to the last"
```

```
} else if ([hat isEqualToString:@"Fedora"]) {
  NSLog(@"Mr. Higgie was an iPhone before there was
iPhone");
} else if ([hat isEqualToString:@"AstronautHelmet"]) {
  NSLog(@"hat is encountered");
} else {
  NSLog(@"Mr. Higgie is currently hatless");
}
//Bloque para el switch y sus diferentes casos, esto es
para enteros
NSInteger day = getDayOfWeek();
switch (day) {
  case 1:
  case 2:
  case 3:
  case 4: {
    [mrHiggie setCurrentHat:@"Fedora"];
    break;
  }
  case 5: {
    [mrHiggie setCurrentHat:@"Sombrero"];
    break;
  }
  case 6:
  case 7: {
    [mrHiggie setCurrentHat:@"AstronautHelmet"];
    break;
  }
}
NSLog(@"Mr. Higgie is wearing: %@", [mrHiggie
currentHat]);

//Enumeración de tipo día, validación con switch. La
enumeración es limitada
DayOfWeek day = getDayOfWeek();
switch (day) {
    case DayOfWeekMonday:
    case DayOfWeekTuesday:
    case DayOfWeekWednesday:
    case DayOfWeekThursday: {
        [mrHiggie setCurrentHat:@"Fedora"];
        break;
    }
    case DayOfWeekFriday: {
        [mrHiggie setCurrentHat:@"Sombrero"];
        break;
```

```
};

for (NSString *word in funnyWords) {
  NSString *definition = funnyWords[word];
  NSLog(@"%@ is defined as %@", word, definition);
}

//for para visualizer los valores de un diccionario
NSDictionary *newHats = @{
  @"Cowboy": @"White",
  @"Conductor": @"Brown",
  @"Baseball": @"Red"
};
for (NSString *hat in newHats){

  NSString *color = newHats[hat];
  NSLog(@"Trying on new %@ %@ hat", color, hat);
  if([mrHiggie tryOnHat:hat withColor:color]) {
    NSLog(@"Mr. Higgie loves it");
  } else {
    NSLog(@"Mr. Higgie hates it");
  }
}

//Block, es el equivalente a un método, puede regresar un valor o no.
void (^myFirstBlock)(void) = ^{
  NSLog(@"Hello from inside the block");
};
myFirstBlock();  //Llamada al block

//Block que acepta valores de tipo string
void (^myFirstBlock)(NSString *) = ^(NSString *mensaje){
  NSLog(@"El mensaje del block es: %@", mensaje);
};
myFirstBlock(@"Hello");  //Llamada al block
myFirstBlock(@"World");  //Llamada al block

//Enumeración en un block usando enumerateObjectUsingBlock
NSArray *newHats = @[@"Cowboy", @"Conductor", @"Baseball",
  @"Beanie", @"Beret", @"Fez"];
[newHats enumerateObjectsUsingBlock:
  ^(NSString *hat, NSUInteger index, BOOL *stop){
    NSLog(@"Trying on hat #%lu: %@", index+1, hat);
  }
];
```

## Cuarto Nivel

```
Telefono  //Crear una clase

```

```
- (NSString *) speak:(NSString *)greeting;
@end
```

```
//Crear la interface y agregar propiedades
@interface Telefono : NSObject
@property NSString *phoneName;
@property NSString *modelNumber;
@end

//Implementación de la clase Persona
#import "Telefono.h"
@implementation Telefono
@end

//Enviar propiedades a la variable phoneName
#import "Telefono.h"
Telefono *talkingiPhone = [[Telefono alloc] init];
talkingiPhone.phoneName = @"Mr. Higgie";
NSLog(@"%@", talkingiPhone.phoneName);

//Implementar la clase, hace una instancia y manda mensajes
#import "Telefono.h"
Telefono *talkingDroid = [[Telefono alloc] init];
talkingDroid.phoneName = @"Android";
talkingDroid.modelNumber = @"Nexus 4";
NSLog(talkingDroid.phoneName);
NSLog(talkingDroid.modelNumber);

//Implementación de un método
@interface Telefono : NSObject
@property NSString *phoneName;
@property NSString *modelNumber;
- (void) speak;
@end


//Uso del método
#import "Telefono.h"
@implementation Telefono
- (void) speak;
{
  NSLog(@"Something to log");
}
@end

//Probar el método
#import "Telefono.h"
Telefono *talkingiPhone = [[Telefono alloc] init];
talkingiPhone.phoneName = @"Mr. Higgie";
[talkingiPhone speak];

//Implementar la variable self que usa la clase
#import "Telefono.h"
@implementation Telefono
```

```

#import "Telefono.h"
@implementation Telefono
- (void)decreaseBatteryLife;
{
  self.batteryLife = @([self.batteryLife intValue] - 1);
}
- (void) reportBatteryLife;
{
    NSLog(@"Battery life is %@", self.batteryLife);
}
- (NSString *)speak:(NSString *)greeting;
{
    NSString *message = [NSString stringWithFormat:@"%@ says %@",
self.phoneName, greeting];
    return message;
}
@end

//Pasando un NSNumber como parámetro
@interface Telefono : NSObject
@property NSString *phoneName;
@property NSString *modelNumber;
@property NSNumber *batteryLife;
- (void) decreaseBatteryLife:(NSNumber *)num;
- (NSString *) speak:(NSString *)greeting;
- (void) reportBatteryLife;
@end

@implementation Telefono
- (void) decreaseBatteryLife:(NSNumber *)num;
{
  self.batteryLife = @([self.batteryLife intValue] - [num
intValue]);
}
- (void) reportBatteryLife;
{
    NSLog(@"Battery life is %@", self.batteryLife);
}
- (NSString *)speak:(NSString *)greeting;
{
    NSString *message = [NSString stringWithFormat:@"%@ says %@",
self.phoneName, greeting];
    return message;
}
@end

//Hacer batteryLife como solo lectura
@interface Telefono : NSObject
@property NSString *phoneName;
@property NSString *modelNumber;
```

```
-(void)speak;
{
    NSLog(@"%@ says Hello There!", self);
}
@end

//Usando self para mostrar el phoneName
#import "Telefono.h"
@implementation Telefono
-(void)speak;
{
    NSLog(@"%@", self.phoneName);
}
@end

//Implementa un método que regresa un mensaje
@interface Telefono : NSObject
@property NSString *phoneName;
@property NSString *modelNumber;

- (NSString *) speak;
@end

#import "Telefono.h"
@implementation Telefono
-(String *)speak
{
    NSString *message = [NSString stringWithFormat:@"%@ says Hello
There!", self.phoneName];
    return message;
}
@end

//Mostrar un mensaje enviado a una clase
#import "Telefono.h"
Telefono *talkingiPhone = [[Telefono alloc] init];
talkingiPhone.phoneName = @"Mr. Higgie";
NSString *phoneMessage = [talkingiPhone speak];
NSLog(@"%@", phoneMessage);

//Pasando argumentos dentro de un método
#import "Telefono.h"
Telefono *talkingiPhone = [[Telefono alloc] init];
talkingiPhone.phoneName = @"Mr. Higgie";
NSLog([talkingiPhone speak:@"I'm sorry"]);
NSLog([talkingiPhone speak:@"I'm bored"]);
NSLog([talkingiPhone speak:@"this stinks"]);

//Define un método con argumentos y lo muestra
@interface Telefono : NSObject
@property NSString *phoneName;
```

```
@property (readonly) NSNumber *batteryLife;
- (void) decreaseBatteryLife:(NSNumber *)arg;
- (NSString *) speak:(NSString *)greeting;
- (void) reportBatteryLife;
@end

#import "Telefono.h"
@implementation Telefono
- (void) decreaseBatteryLife:(NSNumber *)arg;
{
  _batteryLife = @([self.batteryLife intValue] - [arg intValue]);
}
- (void) reportBatteryLife;
{
    NSLog(@"Battery life is %@", self.batteryLife);
}

- (NSString *)speak:(NSString *)greeting;
{
    NSString *message = [NSString stringWithFormat:@"%@ says %@",
self.phoneName, greeting];
    return message;
}
@end

//Asignar una propiedad por default
@interface Telefono : NSObject
@property NSString *phoneName;
@property NSString *modelNumber;
@property (readonly) NSNumber *batteryLife;
- (void) decreaseBatteryLife:(NSNumber *)arg;
- (NSString *) speak:(NSString *)greeting;
- (void) reportBatteryLife;
@end

#import "Telefono.h"

@implementation Telefono

- (Telefono *)init;
{
  self = [super init];
  _batteryLife = @100;
  return [super init];
}
- (void) decreaseBatteryLife:(NSNumber *)arg;
{
  _batteryLife = @([self.batteryLife intValue] - [arg intValue]);
}
- (void) reportBatteryLife;
{
```

```
@property NSString *modelNumber;
@property NSNumber *batteryLife;
- (void)reportBatteryLife:(NSNumber *)batteryLife;
- (NSString *) speak:(NSString *)greeting;
@end

#import "Telefono.h"
@implementation Telefono
- (void)reportBatteryLife:(NSNumber *)batteryLife;
{
    NSLog(@"Battery life is at %@", batteryLife);
}
- (NSString *)speak:(NSString *)greeting;
{
    NSString *message = [NSString stringWithFormat:@"%@ says %@",
self.phoneName, greeting];
    return message;
}
@end

//Acceso a las propiedades usando self
@interface Telefono : NSObject
@property NSString *phoneName;
@property NSString *modelNumber;
@property NSNumber *batteryLife;
- (void) reportBatteryLife;
- (NSString *) speak:(NSString *)greeting;
@end

#import "Telefono.h"
@implementation Telefono
- (void) reportBatteryLife;
{
    NSLog(@"Battery life is %@", self.batteryLife);
}
- (NSString *)speak:(NSString *)greeting;
{
    NSString *message = [NSString stringWithFormat:@"%@ says %@",
self.phoneName, greeting];
    return message;
}
@end

//Metodo para decrementar la bacteria en 1
@interface Telefono : NSObject
@property NSString *phoneName;
@property NSString *modelNumber;
@property NSNumber *batteryLife;
- (void)decreaseBatteryLife;
- (void) reportBatteryLife;
```

```
    NSLog(@"Battery life is %@", self.batteryLife);
}
- (NSString *)speak:(NSString *)greeting;
{
    NSString *message = [NSString stringWithFormat:@"%@ says %@",
self.phoneName, greeting];
    return message;
}
@end

//Crear una variable de instancia
@interface Telefono : NSObject {
  NSNumber *_batteryLife;
}
@property NSString *phoneName;
@property NSString *modelNumber;
- (void) decreaseBatteryLife:(NSNumber *)arg;
- (NSString *) speak:(NSString *)greeting;
- (void) reportBatteryLife;
@end

#import "Telefono.h"

@implementation Telefono
- (Telefono *)init;
{
  self = [super init];
  _batteryLife = @100;
  return [super init];
}
- (void) decreaseBatteryLife:(NSNumber *)arg;
{
  _batteryLife = @([_batteryLife intValue] - [arg intValue]);
}
- (void) reportBatteryLife;
{
    NSLog(@"Battery life is %@", _batteryLife);
}
- (NSString *)speak:(NSString *)greeting;
{
    NSString *message = [NSString stringWithFormat:@"%@ says %@",
self.phoneName, greeting];
    return message;
}
@end
```

## Quinto Nivel

```
//Crea una copia
#import "Telefono.h"
Telefono *talkingiPhone = [[Telefono alloc] init];
talkingiPhone.phoneName = @"Mr. Higgie";
[talkingiPhone decreaseBatteryLife:@5];
Telefono *copy = [talkingiPhone copy];
[copy reportBatteryLife];

//Reponder a un mensaje
#import "Telefono.h"
Telefono *talkingiPhone = [[Telefono alloc] init];
talkingiPhone.phoneName = @"Mr. Higgie";
[talkingiPhone decreaseBatteryLife:@5];
if([talkingiPhone respondsToSelector:@selector(copyWithZone:)]){
  Telefono *copy = [talkingiPhone copy];
  [copy reportBatteryLife];
}

//Haciendo uso de NSCopying para copiar
@interface Telefono : NSObject <NSCopying> {
  NSNumber *_batteryLife;
}
@property NSString *phoneName;
@property NSString *modelNumber;
- (void) decreaseBatteryLife:(NSNumber *)arg;
- (NSString *) speak:(NSString *)greeting;
- (void) reportBatteryLife;
@end

//implementa copyWhitZone
#import "Telefono.h"
@implementation Telefono
- (Telefono *)init;
{
  self = [super init];
  _batteryLife = @100;
  return [super init];
}
- (Telefono *)copyWithZone:(NSZone *)zone;
{
  Telefono *copy = [[Telefono allocWithZone:zone] init];
  return copy;
}
- (void) decreaseBatteryLife:(NSNumber *)arg;
{
  _batteryLife = @([_batteryLife intValue] - [arg intValue]);
}
- (void) reportBatteryLife;
```

```
      NSLog(@"%@'s battery life is %@", self, _batteryLife);
    }

}
- (void) decreaseBatteryLife:(NSNumber *)arg;
{
  _batteryLife = @([_batteryLife intValue] - [arg intValue]);
}
- (NSString *)speak:(NSString *)greeting;
{
    NSString *message = [NSString stringWithFormat:@"%@ says
%@", self.phoneName, greeting];
    return message;
}
@end

//Objetos de clases
#import "Telefono.h"
@implementation Telefono

- (Telefono *)init;
{
  self = [super init];
  _batteryLife = @100;
  return self;
}

- (Telefono *)initWithBatteryLife:(NSNumber *)batteryLife;
{
   _batteryLife = batteryLife;
   return [super init];
}
- (Telefono *)copyWithZone:(NSZone *)zone;
{
  Telefono *copy = [[[self class] allocWithZone:zone]
initWithBatteryLife:_batteryLife];
  copy.phoneName = [NSString stringWithFormat:@"Copy of %@",
self.phoneName];
  return copy;
}
- (void) reportBatteryLife;
{
    if(self.phoneName){
      NSLog(@"%@'s battery life is %@", self.phoneName,
_batteryLife);
    }else{
      NSLog(@"%@'s battery life is %@", self, _batteryLife);
    }
```

```
{
    NSLog(@"%@'s battery life is %@", self.phoneName, _batteryLife);
}
- (NSString *)speak:(NSString *)greeting;
{
    NSString *message = [NSString stringWithFormat:@"%@ says %@",
self.phoneName, greeting];
    return message;
}
@end

//Haciendo uso de nil que es equivalente a 0
#import "Telefono.h"
@implementation Telefono
- (Telefono *)init;
{
  self = [super init];
  _batteryLife = @100;
  return [super init];
}
- (Telefono *) copyWithZone:(NSZone *)zone;
{
   Telefono *copy = [[Telefono allocWithZone:zone] init];
   return copy;
}
- (void) decreaseBatteryLife:(NSNumber *)arg;
{
  _batteryLife = @([_batteryLife intValue] - [arg intValue]);
}
- (void) reportBatteryLife;
{
    if(self.phoneName){
      NSLog(@"%@'s battery life is %@", self.phoneName,
_batteryLife);
    }else{
      NSLog(@"%@'s battery life is %@", self, _batteryLife);
    }
}
- (NSString *)speak:(NSString *)greeting;
{
    NSString *message = [NSString stringWithFormat:@"%@ says %@",
self.phoneName, greeting];
    return message;
}
@end

//Nueva inicialización
#import "Telefono.h"
@implementation Telefono
- (Telefono *)init;
{
```

```

}
- (void) decreaseBatteryLife:(NSNumber *)arg;
{
  _batteryLife = @([_batteryLife intValue] - [arg intValue]);
}
- (NSString *)speak:(NSString *)greeting;
{
    NSString *message = [NSString stringWithFormat:@"%@ says
%@", self.phoneName, greeting];
    return message;
}
@end

//Usando el tipo especial de id
#import "Telefono.h"
@implementation Telefono

- (Telefono *)init;
{
  self = [super init];
  _batteryLife = @100;
  return self;
}
- (Telefono *)initWithBatteryLife:(NSNumber *)batteryLife;
{
   _batteryLife = batteryLife;
   return [super init];
}
- (id)copyWithZone:(NSZone *)zone;
{
  id copy = [[[self class] allocWithZone:zone]
initWithBatteryLife:_batteryLife];
  [copy setPhoneName:[NSString stringWithFormat:@"Copy of %@",
self.phoneName]];
  return copy;
}
- (void) reportBatteryLife;
{
    if(self.phoneName){
      NSLog(@"%@'s battery life is %@", self.phoneName,
_batteryLife);
    }else{
      NSLog(@"%@'s battery life is %@", self, _batteryLife);
    }

}
- (void) decreaseBatteryLife:(NSNumber *)arg;
{
  _batteryLife = @([_batteryLife intValue] - [arg intValue]);
}
```

```
  self = [super init];
  _batteryLife = @100;
  return self;
}
- (Telefono *)initWithBatteryLife:(NSNumber *)batteryLife;
{
   _batteryLife = batteryLife;
   return [super init];
}
- (Telefono *)copyWithZone:(NSZone *)zone;
{
  Telefono *copy = [[Telefono allocWithZone:zone]
initWithBatteryLife:_batteryLife];
  copy.phoneName = [NSString stringWithFormat:@"Copy of %@",
self.phoneName];
  return copy;
}
- (void) reportBatteryLife;
{
    if(self.phoneName){
      NSLog(@"%@'s battery life is %@", self.phoneName,
_batteryLife);
    }else{
```

```
- (NSString *)speak:(NSString *)greeting;
{
    NSString *message = [NSString stringWithFormat:@"%@ says
%@", self.phoneName, greeting];
    return message;
}
@end

//Copia de un objeto
#import "Telefono.h"
Telefono *talkingiPhone = [[Telefono alloc] init];
talkingiPhone.phoneName = @"Mr. Higgie";
Telefono *copy = [talkingiPhone copy];
NSLog(@"The memory address for the original object is %p",
talkingiPhone);
NSLog(@"The memory address for the copied object is %p", copy);
```