Διάλεξη 2: Αντικειμενοστρεφής Προγραμματισμός

Στην ενότητα αυτή θα μελετηθούν τα εξής επιμέρους θέματα:
Εισαγωγή στις έννοιες:

- Προγραμματιστικά Παραδείγματα
- Αντικειμενοστρεφής Προγραμματισμός
- Παραδείγματα

Διδάσκων: Παναγιώτης Ανδρέου

Προγραμματιστικά Παραδείγματα

- Λογικός (Logic) (π.χ., Prolog)
  - Χρησιμοποιεί λογικές προτάσεις για να εκφράσει ένα πρόγραμμα
- Συναρτησιακός (Functional) (π.χ., LISP, OCaml, Erlang)
  - Οι υπολογισμοί δηλώνονται σαν μαθηματικές συναρτήσεις που δεν επηρεάζουν την κατάσταση του προγράμματος
- Προστακτικός (Imperative) (π.χ., FORTRAN, Cobol, Pascal, C)
  - Περιγραφή με δηλώσεις οι οποίες αλλάζουν την κατάσταση του προγράμματος
  - Χρησιμοποιεί μεταβλητές, διαδικασίες και διαδοχική ροή προγράμματος
- Βασισμένος σε συμβάντα (Event-driven) (π.χ., Visual C#)
  - Η ροή του προγράμματος καθορίζεται από συμβάντα (π.χ., mouse click)
- **Αντικειμενοστρεφής (Object Oriented) (π.χ., JAVA, C++)**
  - **Ένα πρόγραμμα μοιάζει με μία λίστα από έργα που πρέπει να εκτελεστούν**
  - **Χρησιμοποιεί Αντικείμενα, Μέθοδοι, Πέρασμα Μηνυμάτων για να εκφράσει ένα πρόγραμμα**

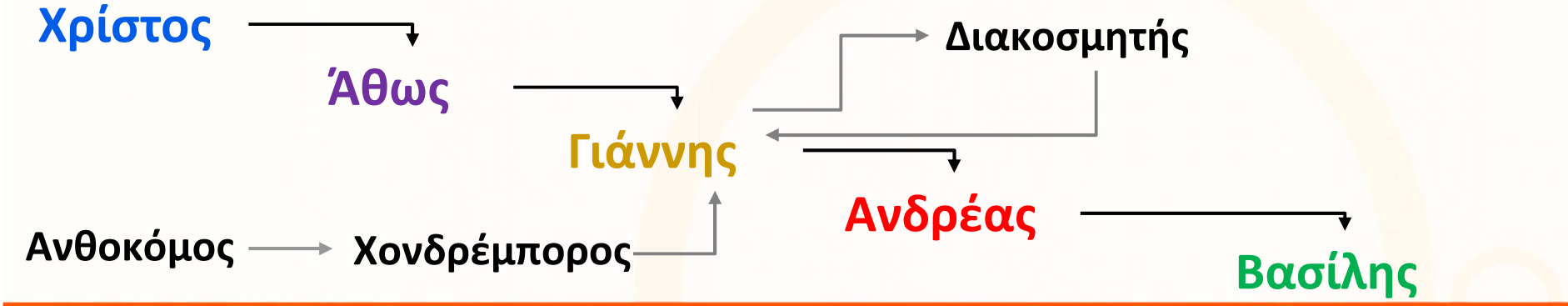
Αντικειμενοστρεφής Προγραμματισμός

- **Τα πάντα είναι αντικείμενα (objects)**
- Κάθε αντικείμενο μπορεί να:
  - Παραλάβει μηνύματα από άλλα αντικείμενα
  - Να επεξεργαστεί δεδομένα
  - Να στείλει μηνύματα σε άλλα αντικείμενα
- Κάθε αντικείμενο είναι μία ανεξάρτητη μηχανή με συγκεκριμένο ρόλο και ευθύνη.
- Οι ενέργειες που μπορεί να εκτελέσει ένα αντικείμενο είναι άμεσα σχετιζόμενες με το αντικείμενο.
- Ένα αντικείμενο μπορεί να περιέχει άλλα αντικείμενα
- Οι κλάσεις οργανώνονται σε δενδρικές δομές μοναδικού πατρικού κόμβου «Ιεραρχία κληρονομικότητας»

Ας δούμε ένα παράδειγμα προβλήματος για να καταλάβουμε καλύτερα τα χαρακτηριστικά/ιδιότητες του αντικειμενοστρεφή προγραμματισμ.

Το παράδειγμα της αποστολής λουλουδιών

- Ο **Χρίστος** θέλει να στείλει λουλούδια στον **Βασίλη**, ο οποίος βρίσκεται σε άλλη πόλη.
- Ο **Χρίστος** δεν μπορεί να παραδώσει τα λουλούδια ο ίδιος, έτσι χρησιμοποιεί τις υπηρεσίες κάποιου ανθοπώλη (**Άθως**).
- Ο **Χρίστος** λέει στον **Άθω** τη διεύθυνση του Βασίλη, το ποσό που μπορεί να διαθέσει και το είδος των λουλουδιών.
- Ο **Άθως** με τη σειρά του επικοινωνεί με άλλο ανθοπώλη (**Γιάννης**) στην πόλη του **Βασίλη**, του μεταφέρει τις πληροφορίες. Ο **Γιάννης** ετοιμάζει την παραγγελία και ενημερώνει τον αποστολέα (**Ανδρέας**).
- Ο **Ανδρέας** παραδίδει τα λουλούδια.



Ιδιότητες Αντικειμενοστρεφή Προγραμματισμού

- Κλάσεις & Στιγμιότυπα
  - Μία κλάση είναι όπως ένα αρχιτεκτονικό σχέδιο: περιγράφει τις ιδιότητες ενός αντικειμένου
  - Από μία κλάση μπορούν να δημιουργηθούν πολλά στιγμιότυπα (αντικείμενα)
  - Κάθε κλάση περιγράφει μία μοναδική οντότητα με ξεχωριστές ιδιότητες στο πρόγραμμα.
  - Παράδειγμα: Ο Άθως και ο Γιάννης είναι και οι δύο ανθοπώληδες

Κλάση

Στιγμιότυπα

```
public class Anthopolis{
…

Anthopolis Giannis;

Anthopolis Athos;
```

Ιδιότητες Αντικειμενοστρεφή Προγραμματισμού

- Ενθυλάκωση (Encapsulation)
  - Περιορίζει την πρόσβαση στα δεδομένα με τροποποιητές πρόσβασης.
  - Παράδειγμα: Το όνομα του Χρίστου μπορεί να το αλλάξει μόνο ο Χρίστος
- Αφαιρετικότητα (Abstraction)
  - Κάθε αντικείμενο έχει συγκεκριμένο ρόλο και ευθύνες.
  - Παράδειγμα: Ο Γιάννης και ο Άθως είναι μόνο ανθοπώληδες, ο Ανδρέας είναι μόνο αποστολέας.
  - Αυτό φυσικά εναπόκειται στις καλές σχεδιαστικές ικανότητες του προγραμματιστή

```
public class Person{
  private name;

  private changeName(String s){
      name=s;
}
…
```

```
public class Anthopolis{
   private createBouquet(int
         noOfFlowers, Color c){
      …
   }
…
public class Apostoleas{
   private sendBouquet(Flowers
f, Address a){
      …
   }
…
```

Ιδιότητες Αντικειμενοστρεφή Προγραμματισμού

- Κληρονομικότητα
  - Περιορίζει την πρόσβαση στα δεδομένα με τροποποιητές πρόσβασης
  - Παράδειγμα: Ο Ανδρέας είναι άτομο, συνεπώς κληρονομεί τα χαρακτηριστικά ενός ατόμου.
- Πολυμορφισμός
  - Υποστηρίζει πολλές μεθόδους με το ίδιο όνομα, αλλά με διαφορετικές παραμέτρους
  - Παράδειγμα: Ο Ανδρέας μπορεί να στείλει τα λουλούδια με διάφορους τρόπους:
    - Με ποδήλατο
    - Με αυτοκίνητο
    - Με αυτοκίνητο σε συγκεκριμένη ώρα

```
public class Person{
  protected name;

 protected changeName(String s){
       name=s;
}
…
public class Anthopolis inherits
Person{
…
```

```
public class Anthopolis{
 private sendFlowers (
       (Flowers x, Bicycle c){…}
 private sendFlowers (
       (Flowers x, Car c){…}
 private sendFlowers (
       (Flowers x, Car c, Time
t){…}
…
```

Πλεονεκτήματα και Μειονεκτήματα της JAVA

ΠΛΕΟΝΕΚΤΗΜΑΤΑ

- Ξεκινάς εύκολα: Α/Σ Γ.Π. με πολλές ομοιότητες με τη C και τη C++ στο συντακτικό.
- Γράφεις λιγότερο κώδικα: μετρήσεις δείχνουν ότι τα προγράμματα JAVA μπορεί να είναι 4 φορές μικρότερα από αντίστοιχα προγράμματα C++.
- Γράφεις καλύτερο κώδικα: ενθαρρύνει τις καλές προγραμματιστικές πρακτικές, αυτόματη διαχείριση μνήμης (αποκομιστής σκυβάλων)
- Γρήγορη κατασκευή κώδικα: απλούστερη ΓΠ από την C++
- Ανεξαρτησία αρχιτεκτονικής - γράφεις κώδικα μια φορά, τρέχει παντού (write once, run anywhere)
- Ευκολότερη εγκατάσταση και διανομή κώδικα.

Πλεονεκτήματα και Μειονεκτήματα της JAVA

ΜΕΙΟΝΕΚΤΗΜΑΤΑ

- Επίδοση:
  - Λόγω της χρήσης του διερμηνέα και της εικονικής μηχανής, η εκτέλεση προγραμμάτων στην πλατφόρμα της JAVA μπορεί να είναι πιό αργή από την εκτέλεση ιθαγενή κώδικα (native code).
  - Ωστόσο, η πρόοδος σε τεχνολογίες μεταγλωττιστών και εικονικών μηχανών, έχει βελτιώσει την επίδοση της JAVA χωρίς να βλάπτει την φορητότητα της γλώσσας.
- Γλώσσα υψηλότερου επιπέδου από τη C/C++:
  - Ο προγραμματιστής έχει πιο περιορισμένες δυνατότητες διαχείρισης μνήμης και σχετικών βελτιστοποιήσεων.
  - Ο προγραμματιστής έχει περιορισμένες δυνατότητες άμεσης πρόσβασης στο υλικό του Η/Υ.

Παραδείγματα: Ένα απλό πρόγραμμα JAVA

```
public class HelloWorld{

  //Αρχικό Σημείο του Προγράμματος
  //Ο ερμηνευτής της JAVA εκτελεί το
  //πρόγραμμα κάνωντας κλήση στην
  //μέθοδο main
  public static void main(String[] args){

    System.out.println("Hello World!!!");

  }

}
```

→ **Παράδειγμα Δήλωσης**

Παραδείγματα: Κλάση

```
class Circle {
  //Η ακτίνα αυτού του κύκλου
  double radius = 1.0;

  //Δημιούργησε ένα αντικείμενο τύπου κύκλος
  Circle () {
  };

  //Δημιούργησε ένα αντικείμενο τύπου κύκλος
  //με συγκεκριμένη ακτίνα
  Circle (double newRadius) {
    radius = newRadius;
  };

  //Επέστρεψε το εμβαδό αυτού του κύκλου
  double getArea () {
    return radius * radius * π;
  }

}
```

Δεδομένα/ Μεταβλητές

Κατασκευαστές

Μέθοδοι

Παραδείγματα: Αντικείμενα

Κλάση Circle

```
class Circle {
  //Η ακτίνα αυτού του κύκλου
  double radius = 1.0;

  //Δημιούργησε ένα αντικείμενο τύπου κύκλος
  Circle () {
  };

  //Δημιούργησε ένα αντικείμενο τύπου κύκλος
  //με συγκεκριμένη ακτίνα
  Circle (double newRadius) {
    radius = newRadius;
  };

  //Επέστρεψε το εμβαδό αυτού του κύκλου
  double getArea () {
    return radius * radius * π;
  }

}
```

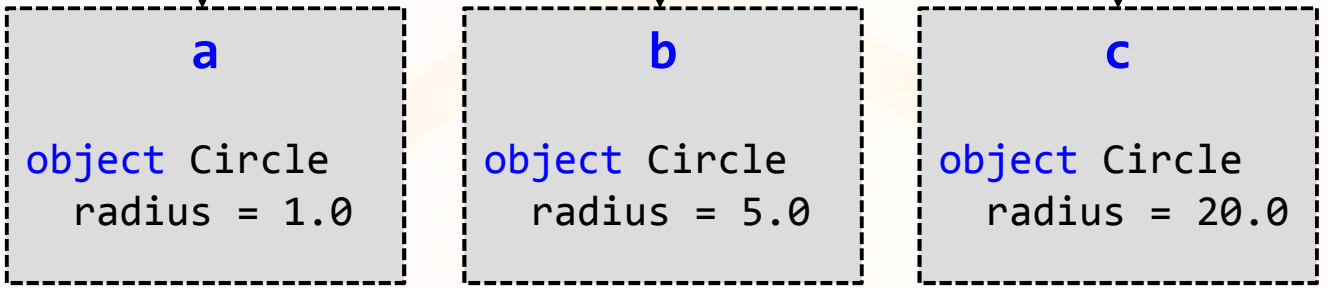
Κλάση Test: περιλαμβάνει την μέθοδο main

```
public class Test{

  public static void main(String[] args){
    Circle a = new Circle();
    Circle b = new Circle(5);
    Circle c = new Circle(20);


…
```

3 αντικείμενα της κλάσης Circle



Αντικείμενα: Κατάσταση, Συμπεριφορά, Ταυτότητα

- Τα αντικείμενα στον ΟΟ προγραμματισμό έχουν τρία βασικά χαρακτηριστικά:
- ***Κατάσταση*** (state) → **Πεδία/Τιμές Δεδομένων**
  Το σύνολο των δεδομένων που αποθηκεύονται στο *εσωτερικό* του αντικειμένου
- ***Συμπεριφορά*** (behavior) → **Λειτουργίες/ Μέθοδοι**
  Το σύνολο των *ενεργειών* που μπορεί να διεκπεραιώσει το αντικείμενο: συναρτήσεις που διαθέτει το αντικείμενο
- ***Ταυτότητα*** (identity) → **Όνομα αντικειμένου/ Διεύθυνση Μνήμης**
  Το "κλειδί" μέσω του οποίου μπορούμε να αποκτήσουμε πρόσβαση στο αντικείμενο

Αλγόριθμοι (ΕΠΛ231)

Τι είναι ένας αλγόριθμος;

Abu Jafar Mohammed ibn Musa Al-Khowarizmi (790-840)

Αλγόριθμος είναι μια πεπερασμένη ακολουθία εντολών, αυστηρά καθορισμένων και εκτελέσιμων σε πεπερασμένο (μετρήσιμο) χρόνο, οι οποίες αν ακολουθηθούν επιτυγχάνεται κάποιο επιθυμητό αποτέλεσμα.

Απαραίτητα κριτήρια:

- Υπάρχει είσοδος και έξοδος
- Καθορισμός εντολών (όχι ασάφειες).
- Περατότητα (να διεκπεραιώνει τον στόχο).

Επίδοση
δεν είναι απαραίτητη αλλά άκρως επιθυμητή

Εξίσωση Wirth: **Αλγόριθμοι + Δεδομένα = Προγράμματα**

Δηλαδή, δοθέντος ενός **προβλήματος**, ένας αλγόριθμος παρέχει τις **οδηγίες** σύμφωνα με τις οποίες τα ***δεδομένα*** του προβλήματος ***μετασχηματίζονται*** και ***συνδυάζονται*** για να προκύψει η **λύση του προβλήματος**.

Παράδειγμα Προγράμματος 1

Δίσεκτο έτος:

- Είσοδος: ένας ακέραιος αριθμός year (έτος)
- Έξοδος: True/False αν είναι δίσεκτος ή όχι
- Αλγόριθμος:

  Ένα έτος είναι δίσεκτο αν

  - Διαιρείται με το 400 ακριβώς
  - Δεν διαιρείτε με το 100 ακριβώς
  - Διαιρείτε με το 4 ακριβώς

```
class LeapYear{
  //μέθοδος ελέγχου δίσεκτου χρόνου
  boolean isLeapYear(int year) {
    boolean result;

    if (year % 400)
      result = true;
    elseif (year % 100)
      result = false;
    elseif(year % 4)
      result = true;
    else
      result = false;

    return result;
  }
…
  public static void main(
    String[] args){
    LeapYear ly = new LeapYear();
    System.out.println(
      ly.isLeapYear(2012) ); }
}
```

Παράδειγμα Προγράμματος 2

Εύρεση ελάχιστης τιμής σε ένα πίνακα:

- Είσοδος: ένας πίνακας με ακέραιους αριθμούς t
- Έξοδος: ο ακέραιος αριθμός με την ελάχιστη τιμή
- Αλγόριθμος:
  - Θέσε το στοιχείο που βρίσκεται στη θέση [0] ως το ελάχιστο min
  - Για κάθε στοιχείο i του πίνακα
    - Αν είναι i είναι μικρότερο του min τότε θέσε min= στοιχείο i
    - Αλλιώς προχώρησε στο επόμενο στοιχείο
  - Επέστρεψε το min

```
class FindMin{
  //μέθοδος ελέγχου δίσεκτου χρόνου
  int isfindMin(int year) {
    int min = array[0];
    for(int i=1; i<array.length; i++){
      if (array[i]<min) min=array[i];
    }
    return min;
  }
…
  public static void main(
    String[] args){

    int array[] = {8,1,5,3,9,6,7};
    FindMin f = new FindMin();
    System.out.println(
       f. isfindMin(array) );
}
```

Παράδειγμα Προγράμματος 3

Εύρεση παραγοντικής τιμής ακέραιου με αναδρομή:

- Παραδείγματα:

  0!=1, 1**!**=1, **2!**=1x2=2, **3!**=1x2x3=6

- Είσοδος: ένας ακέραιος αριθμός x
- Έξοδος: η παραγοντική τιμή του ακέραιου αριθμού x
- Αλγόριθμος:
  - **Αναδρομικός Ορισμός συνάρτησης <factorial>**
    - Βήμα Τερματισμού:
      0! = 1
    - Αναδρομικό Βήμα:
      n!= n x (n-1)!

```
class Factorial{
  //μέθοδος ελέγχου δίσεκτου χρόνου
  int factorial(int x) {
    if (x==0) return 1;
    return x * factorial(x-1);
  }
…
  public static void main(
    String[] args){

    int x=5;
    Factorial f = new Factorial();

    System.out.println(
       f. factorial(x) );
  }
}
```