Fashion_Shows-MFES

December 30, 2017

Contents

1 Designer

```
class Designer

types
 public String = seq of char;
 public DesignerName = String;
 public DesignerAge = String;
 public DesignerNationality = String;
 public DesignerAddress = String;
 public DesignerStyle = String;

instance variables
 public name : DesignerName;
 public age : DesignerAge;
 public nationality : DesignerNationality;
 public address : DesignerAddress;
 public style : DesignerStyle;
 public output : String := "";
 public nr : int := 0;

 operations
  public Designer :
          DesignerName *
```

```
        DesignerAge *
        DesignerNationality *
        DesignerAddress *
        DesignerStyle   ==> Designer
Designer(nm, ag, nt , ad, sty) ==
(
  name := nm;
  age := ag;
  nationality := nt;
  address := ad;
  style := sty;
  return self;
);


--retorna os parametros da class event
public pure getName : () ==> String
  getName() == return name;


 public pure getAge : () ==> String
   getAge() == return age;


public pure getNationality : () ==> String
   getNationality() == return nationality;


public pure getAddress : () ==> String
   getAddress() == return address;


public pure getStyle : () ==> String
   getStyle() == return style;





public printDesigner: () ==> String
printDesigner() == (
output := "Designer Name: "^name^"\n"
     ^"Age: "^age^"\n"
     ^"Nationality: "^nationality^"\n"
     ^"Address: "^address^"\n"
     ^"Style: "^style^"\n";
return output;
);


end Designer
```

Function or operation	Line	Coverage	Calls
Designer	19	100.0%	2
getAddress	45	100.0%	2
getAge	39	100.0%	2
getName	36	100.0%	6
getNationality	42	100.0%	2
getStyle	48	100.0%	2

printDesigner	51	100.0%	1
printModel	51	100.0%	1
toString	52	100.0%	1
Designer.vdmpp		100.0%	19

2 Event

```
class Event

types
 public String = seq of char;
 public EventName = String;
 public EventDate = String;
 public EventLocal = String;
 public EventTime = String;
 public EventDuration = String;
 public EventTheme = String;
 public EventGender = String;
  public EventCollection = String;

instance variables
 public  name : EventName := "";
 public  date : EventDate := "";
 public  local : EventLocal := "";
 public  time : EventTime := "";
 public  duration : EventDuration := "";
 public  theme : EventTheme := "";
 public  gender: EventGender := "";
 public  collection: EventCollection := "";
 public  output: String := "";

 --Lista de desfiles
 private runways: seq of Runway := [];


operations
 public Event :
         EventName *
         EventDate *
         EventLocal *
         EventTime *
         EventDuration *
         EventTheme *
         EventGender *
         EventCollection ==> Event
 Event(nm, dt, lc, hr , dr, tm, gr, cl) == (
  name := nm;
  date := dt;
  local := lc;
  time := hr;
  duration := dr;
  theme := tm;
  gender := gr;
  collection := cl;
  return self
 );


 --retorna os parametros da class event
  public pure getName : () ==> String
```

```
    getName() == return name;


   public pure getDate : () ==> String
     getDate() == return date;


  public pure getLocal : () ==> String
     getLocal() == return local;


  public pure getTime : () ==> String
     getTime() == return time;


  public pure getDuration : () ==> String
     getDuration() == return duration;


  public pure getTheme : () ==> String
     getTheme() == return theme;


  public pure getGender : () ==> String
     getGender() == return gender;


  public pure getCollection : () ==> String
     getCollection() == return collection;


  public pure getRunways : () ==> seq of Runway
     getRunways() == return runways;


  public pure getNumberRunways : () ==> nat
     getNumberRunways() == return len runways;


  public insertRunway : Runway ==> ()
   insertRunway(r) ==
   (
     runways := runways ^  [r];
   );



  public printEvent: () ==> String
  printEvent() == (
  output := "Event Name: "^name^"\n"
       ^"Date: "^date^"\n"
       ^"Time: "^time^"\n"
       ^"Theme: "^theme^"\n"
       ^"Gender: "^gender^"\n"
       ^"Collection: "^collection^"\n";
  return output;
  );

end Event
```

Function or operation	Line	Coverage	Calls

Event	28	100.0%	3
getCollection	71	100.0%	2
getDate	53	100.0%	2
getDuration	62	100.0%	1
getGender	68	100.0%	2
getLocal	56	100.0%	1
getName	50	100.0%	14
getNumberRunways	77	100.0%	3
getRunways	74	100.0%	5
getTheme	65	100.0%	2
getTime	59	100.0%	2
insertRunway	80	100.0%	4
printEvent	86	100.0%	1
printFashionFestival	86	100.0%	1
Event.vdmpp		100.0%	43

3 FashionFestival

```
class FashionFestival

types
 public String = seq of char;
 public FestivalName = String;
 public FestivalDateBegin = String;
 public FestivalDateEnd = String;
 public FestivalLocal = String;

instance variables
 public  name : FestivalName := "";
 public  dateBegin : FestivalDateBegin := "";
 public  dateEnd : FestivalDateEnd := "";
 public  local : FestivalLocal := "";
 private  events: seq of Event := [];
 private  numberEvents: int := 0;
 private  fashionUsers : set of FashionUser := {};
 private output : String := "";


operations
 public FashionFestival :
              FestivalName *
              FestivalDateBegin *
              FestivalDateEnd *
              FestivalLocal
              ==> FashionFestival
  FashionFestival(nm, di, df , lc) ==
  (
    name := nm;
    dateBegin := di;
    dateEnd := df;
    local := lc;
    return self;
  );


  --retorna os parametros da class event
  public pure getName : () ==> String
    getName() == return name;


   public pure getDateBegin : () ==> String
     getDateBegin() == return dateBegin;


  public pure getDateEnd : () ==> String
     getDateEnd() == return dateEnd;


  public pure getLocal : () ==> String
     getLocal() == return local;


  public pure getEvents : () ==> seq of Event
     getEvents() == return events;


 --Adiciona designer ao evento
 public insertEvent : Event ==> ()
  insertEvent(ev) ==
  (
    events := [ev] ^ events;
    numberEvents := numberEvents + 1;
  );



   --Adiciona designer ao evento
 public insertFashionUser: (FashionUser)  ==> ()
  insertFashionUser(us) ==
  (
    fashionUsers := fashionUsers union {us};
  );


 --retorna nr designers do evento
  public pure getNumberEvents : () ==> int
  getNumberEvents() == return numberEvents;


  --retorna utilizadores da aplicaao
  public pure getFashionUsers : () ==> set of FashionUser
  getFashionUsers() == return fashionUsers;


  public pure getNumberFashionUsers: () ==> nat
  getNumberFashionUsers() ==
  return (card fashionUsers);


  public printFashionFestival: () ==> String
  printFashionFestival() == (
  output := "Name: "^name^"\n"
       ^"Date Begin: "^dateBegin^"\n"
       ^"Date End: "^dateEnd^"\n"
       ^"Local: "^local^"\n";
  return output;
  );

end FashionFestival
```

Function or operation	Line	Coverage	Calls
FashionFestival	20	100.0%	2
getDateBegin	39	100.0%	2
getDateEnd	42	100.0%	2
getEvents	48	100.0%	9
getFashionUsers	72	100.0%	2
getLocal	45	100.0%	2
getName	36	100.0%	4
getNumberEvents	68	100.0%	2
getNumberFashionUsers	75	100.0%	1
insertEvent	52	100.0%	4
insertFashionUser	61	100.0%	2
printFashionFestival	79	100.0%	1
FashionFestival.vdmpp		100.0%	33

4 FashionUser

```
class FashionUser

types
 public String = seq of char;
 public Username = String;
 public Password = String;
 public UserName = String;
 public UserAge = String;

instance variables
 private username: Username;
 private password: Password;
 public name: UserName;
 public age: UserAge;
 --designers favoritos
 private  designers: seq of Designer := [];
 private  numberDesigners: int := 0;
  --modelos favoritos
 private  models: seq of Model := [];
 private  numberModels: int := 0;
 --eventos que utilizador vai
 private events: seq of Event := [];
 private  numberEvents: int := 0;
 private output : String := "";


operations
  public FashionUser : Username *
            Password *
            UserName *
            UserAge  ==> FashionUser
  FashionUser(u, p, nm, ag) ==
  (
   username := u;
   password := p;
    name := nm;
    age := ag;
    return self;
  );
```

```
--retorna os parametros da class fashionUSer
public pure getUsername : () ==> String
  getUsername() == return username;


public pure getPassword : () ==> String
  getPassword() == return password;


public pure getName : () ==> String
  getName() == return name;


 public pure getAge : () ==> String
   getAge() == return age;



  --DESIGNERES FAVORITOS

 --Adiciona designer favorito ao utilizador
public insertDesigner : Designer ==> ()
 insertDesigner(d) ==
 (
   numberDesigners := numberDesigners + 1;
   designers := designers ^ [d];
 );


--retorna nr designers favoritos
 public pure getNumberFavDesigners : () ==> nat
 getNumberFavDesigners() == return numberDesigners;


 public getDesigners: () ==> seq of Designer
 getDesigners() == return designers;



 public insertModel : Model ==> ()
 insertModel(d) ==
 (
   numberModels := numberModels + 1;
   models := models ^ [d];
 );


--retorna nr designers favoritos
 public pure getNumberFavModels : () ==> nat
 getNumberFavModels() == return numberModels;


 public getModels: () ==> seq of Model
 getModels() == return models;


  --EVENTOS DO UTILIZADOR

  --Adiciona evento
public insertEvent : Event ==> ()
 insertEvent(ev) ==
 (
   numberEvents := numberEvents + 1;
```

```
      events := events ^ [ev];
    );


  --Retorna nr designers favoritos
   public pure getNumberEvents : () ==> nat
   getNumberEvents() == return numberEvents;


   public getEvents: () ==> seq of Event
   getEvents() == return events;



   public printUser: () ==> String
  printUser() == (
  output := "Username: "^username^"\n"
       ^"Password: "^password^"\n"
       ^"Name: "^name^"\n"
       ^"Age: "^age^"\n";
  return output;
  );

end FashionUser
```

Function or operation	Line	Coverage	Calls
FashionUser	26	100.0%	2
getAge	49	100.0%	2
getDesigners	67	100.0%	2
getEvents	99	100.0%	2
getModels	82	100.0%	2
getName	46	100.0%	2
getNumberEvents	96	100.0%	1
getNumberFavDesigners	64	100.0%	1
getNumberFavModels	79	100.0%	1
getPassword	43	100.0%	2
getUsername	40	100.0%	2
insertDesigner	56	100.0%	2
insertEvent	88	100.0%	2
insertModel	71	100.0%	2
printFashionFestival	102	100.0%	1
printUser	102	100.0%	1
FashionUser.vdmpp		100.0%	27

5 Model

```
class Model

types
 public String = seq of char;
 public ModelName = String;
 public ModelAge = String;
 public ModelNationality = String;
 public ModelAddress = String;

instance variables
 public name : ModelName;
 public age : ModelAge;
 public nationality : ModelNationality;
 public address : ModelAddress;
 public output : String := "";


 operations
  public Model :
          ModelName *
          ModelAge *
          ModelNationality *
          ModelAddress ==> Model
  Model(nm, ag, nt , ad) ==
  (
    name := nm;
    age := ag;
    nationality := nt;
    address := ad;
    return self;
  );


  --retorna os parametros da class Model
  public pure getName : () ==> String
    getName() == return name;


   public pure getAge : () ==> String
     getAge() == return age;


  public pure getNationality : () ==> String
     getNationality() == return nationality;


  public pure getAddress : () ==> String
     getAddress() == return address;



  public printModel: () ==> String
  printModel() == (
  output := "Model Name: "^name^"\n"
       ^"Age: "^age^"\n"
       ^"Nationality: "^nationality^"\n"
       ^"Address: "^address^"\n";
  return output;
  );


end Model
```

Function or operation	Line	Coverage	Calls
Model	17	100.0%	4
getAddress	41	100.0%	2

getAge	35	100.0%	2
getName	32	100.0%	10
getNationality	38	100.0%	2
printModel	44	100.0%	1
printUser	44	100.0%	1
Model.vdmpp		100.0%	22

6 MyTestCase

```
class MyTestCase

operations

  -- Simulates assertion checking by reducing it to pre-condition checking.
  -- If 'arg' does not hold, a pre-condition violation will be signaled.
  -- Verification of pre-conditions must be enabled in order for this to work

  protected assertTrue : bool ==> ()
  assertTrue(arg) == return
  pre arg;

  -- Simulates assertion checking by reducing it to pre-condition checking.
  -- If 'arg' holds, a pre-condition violation will be signaled.
  -- Verification of pre-conditions must be enabled in order for this to work

  protected assertFalse : bool ==> ()
  assertFalse(arg) == return
  pre not arg;

  -- Simulates assertion checking by reducing it to pre-condition checking.
  -- If 'arg' is null or undefined, a pre-condition violation will be signaled.
  -- Verification of pre-conditions must be enabled in order for this to work

  protected assertNotNull : ? ==> ()
  assertNotNull(arg) == return
  pre arg <> nil and arg <> undefined;

  -- Simulates assertion checking by reducing it to post-condition checking.
  -- If values are not equal, prints a message and generates a post-conditions violation.

  protected assertEqual : ? * ? ==> ()
  assertEqual(expected, actual) ==
    if expected <> actual then
    (
      IO`print("Actual value (");
      IO`print(actual);
      IO`print(") different from expected (");
      IO`print(expected);
      IO`println(")\n")
    )
  post expected = actual;

end MyTestCase
```

Function or operation	Line	Coverage	Calls

assertEqual	28	38.8%	0
assertFalse	15	0.0%	0
assertNotNull	22	100.0%	13
assertTrue	8	0.0%	0
MyTestCase.vdmpp		48.3%	13

7 Runway

```
class Runway

types
 public String = seq of char;
 public RunwayName = String;

instance variables
 public  name : RunwayName := "";
 private designers: set of Designer := {};
 private numberDesigners: int := card designers;
 private models: seq of Model := [];
 private numberModels: int := 0;
 private output : String := "";
 operations

 public Runway :
         RunwayName
          ==> Runway
  Runway(nm) == (
  name := nm;
  return self
 );


 public pure getName : () ==> String
    getName() == return name;


 public pure getModels : () ==> seq of Model
     getModels() == return models;


 public pure getDesigners : () ==> set of Designer
     getDesigners() == return designers;


 public pure getDesignersNumber : () ==> nat
     getDesignersNumber() == return card designers;


 public insertDesigner : Designer ==> ()
  insertDesigner(dg) ==
  (

    designers := designers union  {dg};
  );

  --MODELOS DO DESFILE
  --Adiciona model ao desfile

 public insertModel : Model ==> ()
```

```
  insertModel(md) ==
  (
    numberModels := numberModels + 1;
    models := models ^  [md];
  );

  --retorna nr modelso do desfile

  public pure getNumberModels : () ==> int
  getNumberModels() == return numberModels;



  public printRunway: () ==> String
  printRunway() == (
  output := "Runway Name: "^name^"\n";
  return output;
  );


end Runway
```

Function or operation	Line	Coverage	Calls
Runway	15	100.0%	3
getDesigners	29	100.0%	2
getDesignersNumber	32	100.0%	1
getModels	26	100.0%	2
getName	23	100.0%	4
getNumberModels	52	100.0%	1
insertDesigner	35	100.0%	4
insertModel	44	100.0%	5
printModel	55	0.0%	0
printRunway	55	0.0%	0
Runway.vdmpp		81.2%	22

8 TestApp

```
class TestApp

types
public String = seq of char;

instance variables
private  static festivals: seq of FashionFestival := [];
private  static events: seq of Event := [];
private  static designers: seq of Designer := [];
private  static models: seq of Model := [];
private  static users: set of FashionUser := {};
private  static eventsTemp: seq of Event := [];
private  static modelsTemp: seq of Model := [];
private  static designersTemp: seq of Designer := [];
private  static festivalTemp: FashionFestival := new FashionFestival();
private  static runwaysTemp: seq of Runway := [];
operations
```

```
public static printTests: () ==> ()
 printTests() ==
 (
  IO`print("Executing Tests.vdmpp operations...");
    new Tests().run();
   );



 public static getUsers: () ==> set of FashionUser
 getUsers() ==
 (
 return Tests`getAppUsers();
 );


  public static registerUser: (String) *  (String) * (String) * (String) ==> ()
  registerUser(username,password,name,age) ==
  (
   Tests`setAppUser(username,password,name,age);
  );





  public static getFestivals: () ==> seq of FashionFestival
  getFestivals() ==
  (
   return Tests`getFestivals();
  );

  --Nomes dos festivais disponiveis

  public static getFestivalsNames: () ==> ()
 getFestivalsNames() ==
 (
 for counter = 1 to len Tests`getFestivals() do (
    IO`print("\n");
    IO`print(counter);
    IO`print((Tests`getFestivals() (counter)).getName());
    IO`print("\n");
    IO`print("\n");
   );
  );

  --Festival seleccionado apos isto imprimirs

  public static getFestival: (int) ==> FashionFestival
  getFestival(optionFestival) ==
  (
  return (Tests`getFestivals() (optionFestival));
  );

  --eventos do festival selecionado

  public static getFestivalEvents: (int) ==> seq of Event
  getFestivalEvents(optionFestival) ==
  (
  return (Tests`getFestivals() (optionFestival)).getEvents();
  );
  public static getFestivalUsers: (int) ==> set of FashionUser
  getFestivalUsers(optionFestival) ==
  (
  return (Tests`getFestivals() (optionFestival)).getFashionUsers();
  );



  --nomes dos eventos do festival selecionado

  public static getFestivalEventsNames: (int) ==> ()
  getFestivalEventsNames(optionFestival) ==
  (
    IO`print("\"n\"");
    for counter = 1 to len (Tests`getFestivals() (optionFestival)).getEvents() do (
    IO`print(counter);
    IO`print(":  ");
    IO`print(((Tests`getFestivals() (optionFestival)).getEvents() (counter)).getName());
    IO`print("\n");
    );
  );

 --Evento selecionado do festival selecionado

 public static getEvent: (int) * (int) ==> Event
  getEvent(optionFestival,optionEvent) ==
  (
  return (getFestivalEvents(optionFestival) (optionEvent));
  );

 --Retorna desfiles do evento selecionado do festival selecionado

  public static getRunwaysByEvent : (int) * (int) ==> seq of Runway
 getRunwaysByEvent(optionFestival, optionEvent) ==
 (
  return ((getFestivalEvents(optionFestival) (optionEvent)).getRunways());
 );


 public static getRunwaysNames: (int) * (int) ==> ()
  getRunwaysNames(optionFestival,optionEvent) ==
  (
    IO`print("\n");
    for counter = 1 to len getRunwaysByEvent(optionFestival,optionEvent) do (
    IO`print(counter);
    IO`print(":  ");
    IO`print((getRunwaysByEvent(optionFestival,optionEvent) (counter)).getName());
    IO`print("\n");
    );
  );

 --Retorna desfile seleciondo do evento selecionado do festival selecionado

  public static getOneRunwayByEvent : (int) * (int) * (int) ==> Runway
 getOneRunwayByEvent(optionFestival, optionEvent, optionRunWay) ==
 (
  return (getRunwaysByEvent(optionFestival,optionEvent)) (optionRunWay);
 );

 --Retorna modelos do desfile seleciondo do evento selecionado do festival selecionado

 public static getModelsByRunway : (int) * (int) * (int) ==> seq of Model
 getModelsByRunway(optionFestival, optionEvent, optionRunWay) ==
 (
 return getOneRunwayByEvent(optionFestival,optionEvent,optionRunWay).getModels();
```

```
 );



  --Retorna designers do desfile seleciondo do evento selecionado do festival selecionado
  public static getDesignersByRunway : (int) * (int) * (int) ==> set of Designer
 getDesignersByRunway(optionFestival, optionEvent, optionRunWay) ==
 (
 return getOneRunwayByEvent(optionFestival,optionEvent,optionRunWay).getDesigners();
 );





public static main : () ==> ()
  main() ==
  (
   printTests();
  );

end TestApp
```

Function or operation	Line	Coverage	Calls
getDesignersByRunway	145	100.0%	1
getEvent	97	100.0%	1
getFestival	62	100.0%	1
getFestivalEvents	69	100.0%	6
getFestivalEventsNames	84	0.0%	0
getFestivalUsers	75	100.0%	1
getFestivals	42	100.0%	1
getFestivalsNames	49	0.0%	0
getModelsByRunway	130	100.0%	1
getModelsInfsByRunway	136	100.0%	1
getOneRunwayByEvent	123	100.0%	3
getRunwaysByEvent	104	100.0%	4
getRunwaysNames	110	0.0%	0
getUsers	27	100.0%	1
main	154	100.0%	1
printTests	19	100.0%	1
registerUser	33	0.0%	0
TestApp.vdmpp		50.0%	23

9 Tests

```
class Tests is subclass of MyTestCase

types
public String = seq of char;

instance variables
 private static festivals: seq of FashionFestival := [];
 private static events: seq of Event := [];
 private static designers: seq of Designer := [];
 private static models: seq of Model := [];
 private static appUsers: set of FashionUser := {};
  private static runways: seq of Runway := [];


operations


  public run : () ==> ()
  run() ==
  (

    -- VARIABLE DECLARATIONS
   dcl f0: FashionFestival := new FashionFestival("Porto Fashion Week", "04/05/2018", "10/05/2018
       ", "Porto");
   dcl f1: FashionFestival := new FashionFestival("Madrid Weekend", "26/08/2018", "30/08/2018", "
       Madrid");
   dcl ev0: Event := new Event("BaixaShow", "04/05/2018", "Baixa", "12", "3","flores", "Homem", "
       Primavera_Verao");
    dcl ev1: Event := new Event("Fashion Night Out Porto", "10/06/2018","Baixa","20","2","
        GeometricForms","Unisexo","Primavera_Verao");
   dcl ev2: Event := new Event("Black Friday","12/06/2018","Vila do Conde","10","4","Fashion
       Sales","Unisexo","Outono_Inverno");
    dcl d0: Designer:= new Designer("Yves S. L.", "72", "Frances", "Paris", "Classico");
    dcl d1: Designer:= new Designer("Ralph Lauren", "69", "Frances", "Paris", "Classico");
    dcl m0: Model:= new Model("Sara Sampaio", "24", "Portuguesa", "New York");
    dcl m1: Model:= new Model("Claudia Schiffer","47","Alem","Alemanha");
   dcl m2: Model:= new Model("Naomi Campbell","47","Inglesa","Inglaterra");
   dcl m3: Model:= new Model("Kate Moss","43","Inglesa","Inglaterra");
   dcl u0: FashionUser:= new FashionUser("Joao","1234","Joao", "30");
   dcl u1: FashionUser:= new FashionUser("Maria","1234","Maria", "34");
   dcl r0: Runway := new Runway("Meet winter collecion");
   dcl r1: Runway := new Runway("African Power");
   dcl r2: Runway := new Runway("Nautical Vibes");

   r0.insertDesigner(d0);
   r0.insertDesigner(d1);
   r0.insertModel(m0);

   r1.insertDesigner(d1);
   r1.insertModel(m1);
   r1.insertModel(m2);
   r1.insertModel(m3);

   r2.insertDesigner(d0);
   r2.insertModel(m2);


   festivals := festivals ^ [f0];
   festivals := festivals ^ [f1];
   events := events ^ [ev0];
   events := events ^ [ev1];
   events := events ^ [ev2];
   designers := designers ^ [d0];
   designers := designers ^ [d1];
   models := models ^ [m0];
   models := models ^ [m1];
   models := models ^ [m2];
   models := models ^ [m3];
   appUsers := appUsers union {u0};
   appUsers := appUsers union {u1};
   runways := runways ^ [r0];
   runways := runways ^ [r1];
   runways := runways ^ [r2];

   ev0.insertRunway(r0);
   ev1.insertRunway(r1);
   ev1.insertRunway(r2);
   ev2.insertRunway(r2);

    assertNotNull(f0);
    assertNotNull(f1);

    assertNotNull(ev0);
    assertNotNull(ev1);
    assertNotNull(ev2);

    assertNotNull(d0);
    assertNotNull(d1);

    assertNotNull(m0);
    assertNotNull(m1);
    assertNotNull(m2);
    assertNotNull(m3);

    assertNotNull(u0);
    assertNotNull(u1);

    -- EXECUTE Fashion Festival
    assertEqual("Porto Fashion Week", f0.getName());
    assertEqual("04/05/2018", f0.getDateBegin());
    assertEqual("10/05/2018", f0.getDateEnd());
    assertEqual("Porto", f0.getLocal());

    f0.insertEvent(ev0);
    f0.insertEvent(ev2);
    f1.insertEvent(ev0);
    f1.insertEvent(ev1);
    f0.insertFashionUser(u0);
    f0.insertFashionUser(u1);

  assertEqual(2,f0.getNumberEvents());

  assertEqual(ev0.getName(),((f0.getEvents() (2)).getName()));
  assertEqual(2,f1.getNumberEvents());
  assertEqual(ev1.getName(),((f1.getEvents() (1)).getName()));
    assertEqual(ev1.getName(),((f1.getEvents() (1)).getName()));
    assertEqual(2, card f0.getFashionUsers());
    assertEqual(2,f0.getNumberFashionUsers());

    assertEqual(("Name: "^f0.getName()^"\n"
           ^"Date Begin: "^f0.getDateBegin()^"\n"
           ^"Date End: "^f0.getDateEnd()^"\n"
           ^"Local: "^f0.getLocal()^"\n")
           ,f0.printFashionFestival());



    -- EXECUTE EVENT
    assertEqual("BaixaShow", ev0.getName());
    assertEqual("04/05/2018", ev0.getDate());
    assertEqual("Baixa", ev0.getLocal());
    assertEqual("12", ev0.getTime());
    assertEqual("3", ev0.getDuration());
    assertEqual("flores", ev0.getTheme());
    assertEqual("Homem", ev0.getGender());
    assertEqual("Primavera_Verao", ev0.getCollection());
```

```
    /*
    ev0.insertRunway(r0);
   ev1.insertRunway(r1);
   ev1.insertRunway(r2);
   ev2.insertRunway(r2);
   */

   assertEqual(1,ev0.getNumberRunways());
   assertEqual(2,ev1.getNumberRunways());
   assertEqual(1,ev2.getNumberRunways());
   assertEqual(r0.getName(),(ev0.getRunways() (1)).getName());

   assertEqual(("Event Name: "^ev0.getName()^"\n"
       ^"Date: "^ev0.getDate()^"\n"
       ^"Time: "^ev0.getTime()^"\n"
       ^"Theme: "^ev0.getTheme()^"\n"
       ^"Gender: "^ev0.getGender()^"\n"
       ^"Collection: "^ev0.getCollection()^"\n")
           ,ev0.printEvent());
    -- EXECUTE Runway
    /*
    r0.insertDesigner(d0);
   r0.insertDesigner(d1);
   r0.insertModel(m0);
   r1.insertDesigner(d1);
   r1.insertModel(m1);
   r1.insertModel(m2);
   r1.insertModel(m3);
   r2.insertDesigner(d0);
   r2.insertModel(m2);
   */

   assertEqual(r0.getDesignersNumber(),card r0.getDesigners());
  assertEqual(1,r0.getNumberModels());
  assertEqual(m0.getName(),(r0.getModels() (1)).getName());

    -- EXECUTE DESIGNER
    assertEqual("Yves S. L.", d0.getName());
    assertEqual("72", d0.getAge());
    assertEqual("Frances", d0.getNationality());
    assertEqual("Paris", d0.getAddress());
    assertEqual("Classico", d0.getStyle());

    assertEqual(("Designer Name: "^d0.getName()^"\n"
       ^"Age: "^d0.getAge()^"\n"
       ^"Nationality: "^d0.getNationality()^"\n"
       ^"Address: "^d0.getAddress()^"\n"
       ^"Style: "^d0.getStyle()^"\n")
           ,d0.printDesigner());


    -- EXECUTE MODEL
    assertEqual("Sara Sampaio", m0.getName());
    assertEqual("24", m0.getAge());
    assertEqual("Portuguesa", m0.getNationality());
    assertEqual("New York", m0.getAddress());

     assertEqual(("Model Name: "^m0.getName()^"\n"
       ^"Age: "^m0.getAge()^"\n"
       ^"Nationality: "^m0.getNationality()^"\n"
       ^"Address: "^m0.getAddress()^"\n")
           ,m0.printModel());

    -- FASHION USER
    assertEqual("Joao", u0.getName());
```

```
 assertEqual("30", u0.getAge());
 assertEqual("Joao", u0.getUsername());
 assertEqual("1234", u0.getPassword());

 assertEqual(("Username: "^u0.getUsername()^"\n"
    ^"Password: "^u0.getPassword()^"\n"
    ^"Name: "^u0.getName()^"\n"
    ^"Age: "^u0.getAge()^"\n")
        ,u0.printUser());

 u0.insertEvent(ev0);
 u0.insertEvent(ev1);
 assertEqual(2,u0.getNumberEvents());
 assertEqual(ev0.getName(), (u0.getEvents() (1)).getName());
 assertEqual(ev1.getName(), (u0.getEvents() (2)).getName());

  u0.insertDesigner(d0);
  u0.insertDesigner(d1);
  assertEqual(2,u0.getNumberFavDesigners());
  assertEqual(d0.getName(), (u0.getDesigners() (1)).getName());
  assertEqual(d1.getName(), (u0.getDesigners() (2)).getName());

 u0.insertModel(m0);
 u0.insertModel(m1);
  assertEqual(2,u0.getNumberFavModels());
  assertEqual(m0.getName(), (u0.getModels() (1)).getName());
  assertEqual(m1.getName(), (u0.getModels() (2)).getName());

 /*
 f0.insertEvent(ev0);
 f0.insertEvent(ev2);
 f1.insertEvent(ev0);
 f1.insertEvent(ev1);
 f0.insertFashionUser(u0);
 f0.insertFashionUser(u1);

 ev0.insertRunway(r0);
ev1.insertRunway(r1);
ev1.insertRunway(r2);
ev2.insertRunway(r2);

 r0.insertDesigner(d0);
r0.insertDesigner(d1);
r0.insertModel(m0);
r1.insertDesigner(d1);

r1.insertModel(m1);
r1.insertModel(m2);
r1.insertModel(m3);

r2.insertDesigner(d0);
r2.insertModel(m2);
*/


  --TestApp
   assertEqual(2, card TestApp`getUsers());

   assertEqual(2, len TestApp`getFestivals());
   assertEqual(f0.getName(), TestApp`getFestival(1).getName());
  assertEqual(2, len TestApp`getFestivalEvents(1));

  assertEqual(2, card TestApp`getFestivalUsers(1));
  assertEqual(ev2.getName(),TestApp`getEvent(1,1).getName());
 assertEqual(1,len TestApp`getRunwaysByEvent(1,1));
```

```
    assertEqual(r2.getName(),TestApp`getOneRunwayByEvent(1,1,1).getName());
    assertEqual(m2.getName(),(TestApp`getModelsByRunway(1,1,1) (1)).getName());
    assertEqual(1, card TestApp`getDesignersByRunway(1,1,1));

    assertEqual(2, len getFestivals());
    assertEqual(3, len getEvents());
    assertEqual(2, len getDesigners());
    assertEqual(4, len getModels());
    assertEqual(3, len getRunways());
    assertEqual(2, card getAppUsers());
  );

  public  static getFestivals : () ==> seq of FashionFestival
  getFestivals() == return festivals;

  public  static getEvents : () ==> seq of Event
  getEvents() == return events;

  public  static getDesigners : () ==> seq of Designer
  getDesigners() == return designers;

 public  static getModels : () ==> seq of Model
  getModels() == return models;

  public  static getAppUsers : () ==> set of FashionUser
  getAppUsers() == return appUsers;

  public  static getRunways : () ==> seq of Runway
  getRunways() == return runways;

  public static setAppUser: (String) *  (String) * (String) * (String) ==> ()
  setAppUser(username,password,name,age) ==
  (
   dcl u5 : FashionUser:= new FashionUser(username,password,name,age);
   appUsers := appUsers union {u5};
   );

end Tests
```

Function or operation	Line	Coverage	Calls
getAppUsers	254	100.0%	2
getDesigners	248	100.0%	1
getEvents	245	100.0%	1
getFestivals	242	100.0%	10
getModels	251	100.0%	1
getRunways	257	100.0%	1
run	17	100.0%	3
setAppUser	260	0.0%	0
Tests.vdmpp		98.4%	19