ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΕΧΝΟΛΟΓΙΑ ΛΟΓΙΣΜΙΚΟΥ – ΗΥ352

ΧΕΙΜΕΡΙΝΟ ΕΞΑΜΗΝΟ 2023
ΔΙΔΑΣΚΩΝ: ΑΝΤΩΝΙΟΣ ΣΑΒΒΙΔΗΣ

ΕΡΓΑΣΙΑ (ομάδες δύο ατόμων)

Ανάθεση: Τρίτη 5 Δεκεμβρίου 2023

Παράδοση: Πριν την τελική εξέταση (ακριβής ημερομηνία θα ανακοινωθεί όταν θα πλησιάζουμε σε εκείνη την ημερομηνία)

Θέμα – Κατασκευή γλώσσας για προσομοίωση μάχης Pokemon

Θέμα της εργασίας είναι η κατασκευή μιας γλώσσας ειδικού σκοπού μέσα από την οποία ο προγραμματιστής θα μπορεί να προσομοιώσει μάχες εμπνευσμένες από το κινούμενο σχέδιο Pokémon. Η γλώσσα που θα κατασκευάσετε (περιγράφεται παρακάτω) θα γίνεται compiled ως C++ οπότε θα χρειαστείτε ένα ή περισσότερα header files με κατάλληλους ορισμούς ώστε το πρόγραμμά σας που είναι γραμμένο στη γλώσσα σας να αντιστοιχεί σε valid C++ κώδικα και να κάνει compile σωστά. Όπως και στη C++ τα whitespaces θα αγνοούνται. Ένα πρόγραμμα λοιπόν θα **πρέπει** να έχει την εξής μορφή:

```
#include < PokemonLeague.h>

BEGIN_GAME
        …
END_GAME
```

Στοιχεία της γλώσσας

Ορισμός pokemons

Η δήλωση pokemons γίνεται με τα παρακάτω συντακτικά (προσέξτε ότι στο τέλος κάθε δήλωσης δεν υπάρχει semicolon):

```
CREATE POKEMON {
      NAME: "name",
      TYPE: "type",
      HP: health_points
}

CREATE POKEMONS [
      POKEMON {
            NAME: "name",
            TYPE: "type",
            HP: health_points
      },
      POKEMON {
            NAME: "name",
            TYPE: "type",
            HP: health_points
      },

      ...
]
```

Το *name* είναι το αναγνωριστικό του pokemon ώστε να μπορούμε να αναφερθούμε σε αυτό στη συνέχεια, το *type* δείχνει τον τύπο του pokemon *(Electric, Fire, Water, Grass)* και το *health_points* είναι η ζωή που θα έχει το pokemon. Παρακάτω φαίνονται 3 παραδείγματα για την δημιουργία pokemon.

Προσοχή: Τα ονόματα των pokemon θα **μπορούν** να περιέχουν και κενό.

Παραδείγματα:

```
CREATE POKEMON {
	NAME: "Pikachu",
	TYPE: "Electric",
	HP: 120
}

CREATE POKEMON {
	NAME: "Squirtle",
	TYPE: "Water",
	HP: 100
}

CREATE POKEMONS [
	POKEMON{
		NAME: "Ho Oh",
		TYPE: "Fire",
		HP: 120
	},
	POKEMON{
		NAME: "Bulbasaur",
		TYPE: "Grass",
		HP: 85
	}
]
```

Ορισμός ability

Η δήλωση ενός ability γίνεται με τα παρακάτω συντακτικά (προσέξτε ότι στο τέλος κάθε δήλωσης δεν υπάρχει semicolon):

```
CREATE ABILITY {
	NAME: "ability_name",
	ACTION: START
		...
	END
}

CREATE ABILITIES [
	ABILITY {
		NAME: "ability_name",
		ACTION: START
			...
		END
	},
	ABILITY {
		NAME: "ability_name",
		ACTION: START
			...
		END
	},

	...
]
```

Το *ability_name* είναι το αναγνωριστικό του ability ώστε να μπορούμε να αναφερθούμε σε αυτό στη συνέχεια.

Προσοχή: Τα ονόματα των abilities **δεν θα μπορούν** να περιέχουν κενό.

- ***Action***

  Περιέχει τον κώδικα που θα εκτελεστεί όταν κάποιο pokemon χρησιμοποιήσει το συγκεκριμένο ability. Μπορούμε να αναφερθούμε σε αυτόν που κάνει το ability με το keyword **ATTACKER** και σε αυτόν που δέχεται το ability με το keyword **DEFENDER**.

  Οι εντολές που μπορούν να γραφτούν μέσα στο Action ενός ability είναι οι εξής:

  - Εντολή **`DAMAGE DEFENDER/ATTACKER`** *number* κατά την οποία ένα pokemon κάνει damage είτε στον εαυτό του (ATTACKER) είτε στο αντίπαλο pokemon (DEFENDER).

  - Εντολή **`HEAL DEFENDER/ATTACKER`** *number* κατά την οποία ένα pokemon κάνει heal είτε στον εαυτό του (ATTACKER) είτε στο αντίπαλο pokemon (DEFENDER).

  - Εντολή **`POKEBALL DEFENDER/ATTACKER`** *pokeball_value* όπου:
    - `POKEBALL DEFENDER` **`---α`** το defender pokemon βγαίνει από το pokeball
    - `POKEBALL DEFENDER` **`_`** το defender pokemon επιστρέφει στο pokeball
    - `POKEBALL ATTACKER` **`---α`** το attacker pokemon βγαίνει από το pokeball
    - `POKEBALL ATTACKER` **`_`** το attacker pokemon επιστρέφει στο pokeball

  - Εντολή **`GET_HP(DEFENDER/ATTACKER)`** η οποία επιστρέφει το HP του αντιπάλου (DEFENDER) ή του εαυτού του (ATTACKER).

  - Εντολή **`GET_TYPE(DEFENDER/ATTACKER)`** η οποία επιστρέφει το type που ανήκει το αντίπαλο pokemon (DEFENDER) ή το ίδιο (ATTACKER).

  - Εντολή **`GET_NAME(DEFENDER/ATTACKER)`** η οποία επιστρέφει το όνομα του αντιπάλου (DEFENDER) ή του εαυτού του (ATTACKER).

  - Εντολή **`IS_IN_POKEBALL (DEFENDER/ATTACKER)`** η οποία επιστρέφει αν το αντίπαλο pokemon (DEFENDER) ή ο ίδιος (ATTACKER) έχει μπει στο pokeball του.

Σύγκριση, λογικές πράξεις και εντολές ελέγχου μέσα στο Action :

- Συγκρίσεις μεταξύ τιμών (**==, !=, >, >=, <, <=**)
  Παραδείγματα:
  - `GET_TYPE(DEFENDER) == "Fire"`
  - `GET_HP(ATTACKER) <= 60`

- Λογικές πράξεις (**`AND, OR, NOT`**), όπου το *AND, OR* έχει **τουλάχιστον** 2 στοιχεία ενώ το *NOT* έχει **ένα**.

Παραδείγματα:

- `AND (GET_TYPE(ATTACKER) == "Electric", GET_HP(ATTACKER) > 20)`
- ```
  OR ( GET_TYPE(DEFENDER) == "Fire",
       NOT (GET_TYPE(ATTACKER) == "Electric"),
       GET_HP(DEFENDER) <= 20
     )
  ```
- `NOT (AND (GET_HP(DEFENDER) > 20, GET_HP(DEFENDER) < 70))`

- Εντολή **IF** *Condition* **DO** *Commands* **ELSE_IF** *Condition* **DO** *Commands* **ELSE** *Commands* **END**.

  Παραδείγματα:

  ```
  IF AND (GET_HP(DEFENDER) > 30, GET_HP(DEFENDER) < 70) DO
        DAMAGE DEFENDER 20
  END
  ```

  ```
  IF GET_HP(DEFENDER) <= 20 DO
        DAMAGE DEFENDER 10
  ELSE_IF GET_HP(DEFENDER) <= 50 DO
        DAMAGE DEFENDER 20
  ELSE
        DAMAGE DEFENDER 30
  END
  ```

  ```
  IF GET_HP(ATTACKER) <= 30 DO
        IF GET_TYPE(ATTACKER)!= "Fire" DO
              HEAL ATTACKER 20
        ELSE
              HEAL ATTACKER 30
        END
  END
  ```

- Εντολή επανάληψης **FOR** *rounds_number* **ROUNDS DO** *Commands* **END** *η* οποία εκτελεί τα *Commands* για τα επόμενα *rounds_number* rounds.

  Παράδειγμα:

  ```
  FOR 5 ROUNDS DO
        IF AND (GET_HP(DEFENDER) > 10, GET_HP(DEFENDER) < 80) DO
              DAMAGE DEFENDER 20
        END
  END
  ```

- Εντολή **AFTER** *rounds number* **ROUNDS DO** *Commands* **END** η οποία εκτελεί τα *Commands* μετά από *round_number* rounds.

  Παράδειγμα:

  ```
  AFTER 2 ROUNDS DO
        POKEBALL ATTACKER ---α
  END
  ```

- Εντολή **SHOW** *output* η οποία εκτυπώνει τα *output* στην κονσόλα.

  Παραδείγματα:

  - `SHOW GET_HP(ATTACKER)`
  ```

- ```
  SHOW "Name: " << GET_NAME(DEFENDER)<< "Type: " <<
  GET_TYPE(DEFENDER)
  ```

Παραδείγματα για δημιουργία Abilities (μαζί με Actions):

```
CREATE ABILITY {
	NAME: "Electric_Shock",
	ACTION: START
		IF GET_HP(ATTACKER) < 30 DO
			HEAL ATTACKER 25
		ELSE
			HEAL ATTACKER 15
		END
	END
}

CREATE ABILITY {
	NAME: "Solar_Power",
	ACTION: START
		POKEBALL ATTACKER ---α
		DAMAGE DEFENDER 20
	END
}

CREATE ABILITIES [
	ABILITY {
		NAME: "Electric_Shock",
		ACTION: START
			IF GET_HP (ATTACKER) < 30 DO
				HEAL ATTACKER 25
			ELSE
				HEAL ATTACKER 15
			END
		END
	},
	ABILITY {
		NAME: "Blaze",
		ACTION: START
			DAMAGE DEFENDER 22
		END
	}
]
```

Εκμάθηση ability σε pokemon

Ένα pokemon για να μπορέσει να επιτεθεί χρειάζεται να έχει κάποια abilities. Η εκμάθηση ενός ability στο pokemon γίνεται με το παρακάτω συντακτικό (προσέξτε ότι στο τέλος κάθε δήλωσης δεν υπάρχει semicolon):
```

```
DEAR "pokemon name" LEARN [
      ABILITY_NAME(ability_name)
      ABILITY_NAME(ability_name)
      ...
]
```

Το *pokemon name* αντιστοιχεί στο όνομα του pokemon που θέλουμε να μάθει τα abilities, το *ability name* αντιστοιχεί στο όνομα του ability.

Προσοχή: Το *ability name* δεν περικλείεται από τα σύμβολα "". Επίσης τα στοιχεία της λίστας των ABILITY_NAME **ΔΕΝ** χωρίζονται με κόμμα.

Παραδείγματα:

```
DEAR "Pikachu" LEARN [
      ABILITY_NAME(Electric_Shock)
      ABILITY_NAME(Lightning_Rod)
]

DEAR "Ho Oh" LEARN [
      ABILITY_NAME(Solar_Power)
      ABILITY_NAME(Blaze)
      ABILITY_NAME(Tough_Claws)
      ABILITY_NAME(Drought)
]
```

Types

Κάθε pokemon ανήκει σε κάποιο συγκεκριμένο type. Λόγω αντιπαλότητας ανάμεσα στα types τα pokemon έχουν αναπτύξει κάποιες ειδικές δυνάμεις για να μπορούν να ανταποκριθούν στη μάχη.

Πιο συγκεκριμένα:

- Τα pokemon που ανήκουν στο type **Electric** δέχονται 30% **λιγότερο** damage από pokemon του **Fire** και 20% **λιγότερο** damage από τα **άλλα types**.
- Τα pokemon που ανήκουν στο type **Fire** κάνουν 20% **περισσότερο** damage σε pokemon του **Electric** και 15% **περισσότερο** damage σε pokemon των **άλλων types**.
- Τα pokemon που ανήκουν στο type **Water** κάνουν 7% **περισσότερο** damage σε όλους και δέχονται 7% **λιγότερο** damage από όλα τα pokemon.
- Τα pokemon που ανήκουν στο type **Grass** στα rounds που είναι **περιττοί** αριθμοί κάνουν 7% **περισσότερο** damage σε όλους και στα rounds που είναι **άρτιοι** αριθμοί δέχονται **αυτόματα** heal 5% του max health τους κατά την έναρξη του round.

Η Μάχη

Η μάχη μεταξύ δύο pokemon γίνεται από την **κονσόλα** και αναπαρίσταται από την εντολή **DUEL**

Κατά την έναρξη της μάχης:

- Θα πρέπει να εμφανίζεται η λίστα με τα pokemon που έχουν δηλωθεί
- Οι 2 παίκτες επιλέγουν από την λίστα το pokemon που θέλουν να χειρίζονται
- Προσοχή, μπορούν και οι δύο παίκτες να επιλέξουν το **ίδιο** pokemon
- Κατά την έναρξη της μάχης και τα δυο pokemon βγαίνουν από τα pokeballs.

Κατά τη διάρκεια ενός γύρου:

- Θα πρέπει να εμφανίζεται η λίστα με τα διαθέσιμα abilities του pokemon που είναι η σειρά του, από την οποία θα επιλέγει το ability του. Αντίστοιχα, και για τον δεύτερο παίκτη.
  Διαθέσιμα είναι όλα τα abilities που έχει μάθει το pokemon (με την εντολή LEARN) κατά τη διάρκεια του παιχνιδιού.
  Σε περίπτωση που κάποιο pokemon δεν έχει βγει από το pokeball δεν χρειάζεται το παραπάνω αλλά πρέπει να τυπώνεται κατάλληλο μήνυμα (δες Screenshot).
- Αφού ένα pokemon εκτελέσει το ability του θα πρέπει να εκτυπώνεται στην κονσόλα η κατάσταση και των 2 pokemon. π.χ.

  Name: *Pikachu*
  HP: *100*
  Type: *Electric*

Η μάχη τερματίζεται με την ήττα ενός από τα δύο pokemon (έχει χαθεί όλο το HP), που αμέσως αναδεικνύει το αντίπαλο pokemon σε νικητή.

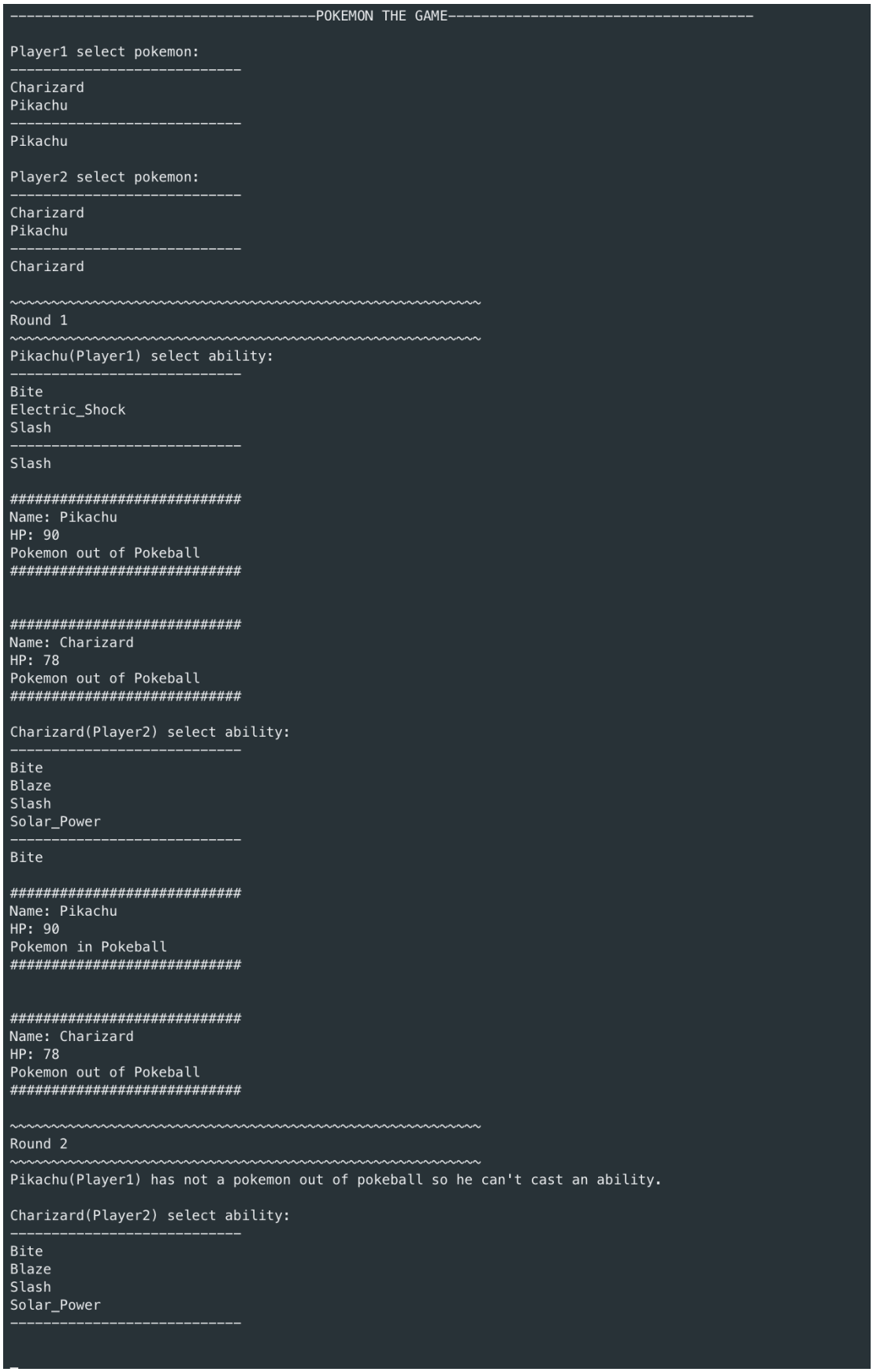
Ακολουθεί **ολοκληρωμένο παράδειγμα** μάχης, Screenshot με το **output** που **ΠΡΕΠΕΙ** να ακολουθήσετε καθώς και μερικά **hints**.

Ολοκληρωμένο Παράδειγμα:

```
#include "PokemonLeague.h"
BEGIN_GAME

    CREATE ABILITY {
        NAME: "Bite",
        ACTION: START
            //Επιστρέφει το αντίπαλο pokemon στο pokeball
            POKEBALL DEFENDER _
              // Μετά απο 2 γύρους ξαναβγάζει τον αντίπαλο(DEFENDER)
              AFTER 2 ROUNDS DO
                   POKEBALL DEFENDER ---α
            END
```

```
        END
    }

    CREATE ABILITY {
        NAME: "Solar_Power",
        ACTION: START
            // Για 5 γύρους κάνει 8 damage στον αντίπαλο (DEFENDER)
            FOR 5 ROUNDS DO
                DAMAGE DEFENDER 8
            END
        END
    }

    CREATE ABILITY {
        NAME: "Slash",
        ACTION: START
              // Κάνει 22 damage στον αντίπαλο (DEFENDER)
              DAMAGE DEFENDER 22
        END
    }

     CREATE ABILITY {
      NAME: "Blaze",
      ACTION: START
          // Κάνει heal στον εαυτό του (ATTACKER)
          HEAL ATTACKER 30
      END
      }

    CREATE POKEMON {
        NAME: "Charizard",
        TYPE: "Fire",
        HP: 100
    }

    CREATE POKEMON {
        NAME: "Pikachu",
        TYPE: "Electric",
        HP: 90
    }

    DEAR "Charizard" LEARN [
        ABILITY_NAME(Bite)
        ABILITY_NAME(Solar_Power)
        ABILITY_NAME(Slash)
        ABILITY_NAME(Blaze)
   ]

    DEAR "Pikachu" LEARN [
        ABILITY_NAME(Bite)
        ABILITY_NAME(Slash)
        ABILITY_NAME(Electric_Shock)
    ]

    DUEL

END_GAME
```

Screenshot από παράδειγμα μάχης:



```
-----------------------------------POKEMON THE GAME-----------------------------------

Player1 select pokemon:
---------------------------
Charizard
Pikachu
---------------------------
Pikachu

Player2 select pokemon:
---------------------------
Charizard
Pikachu
---------------------------
Charizard

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Round 1
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Pikachu(Player1) select ability:
---------------------------
Bite
Electric_Shock
Slash
---------------------------
Slash

##########################
Name: Pikachu
HP: 90
Pokemon out of Pokeball
##########################


##########################
Name: Charizard
HP: 78
Pokemon out of Pokeball
##########################

Charizard(Player2) select ability:
---------------------------
Bite
Blaze
Slash
Solar_Power
---------------------------
Bite

##########################
Name: Pikachu
HP: 90
Pokemon in Pokeball
##########################


##########################
Name: Charizard
HP: 78
Pokemon out of Pokeball
##########################

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Round 2
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Pikachu(Player1) has not a pokemon out of pokeball so he can't cast an ability.

Charizard(Player2) select ability:
---------------------------
Bite
Blaze
Slash
Solar_Power
---------------------------

_
```

Hints

Για να μετατρέψετε το συντακτικό της γλώσσας σε valid C++ χρησιμοποιήστε:

- Τη δυνατότητα για operator overloading που σας προσφέρει η C++, δίνοντας μεγάλη προσοχή στην προτεραιότητα των operators.
  - operator[]
    - Για την εκμάθηση ικανοτήτων σε pokemon.
  - operator,
    - Για να μαζεύετε εκφράσεις που έχουν κόμμα ανάμεσα τους.

- Δημιουργία προσωρινών στιγμιότυπων ως επιστρεφόμενα αποτελέσματα, αλλά και ως βοηθητικά στιγμιότυπα σε εκφράσεις.

- Αρκετά τον preprocessor αφού λέξεις κλειδιά της γλώσσας όπως *CREATE, POKEMON, DAMAGE,* κτλ. θα είναι macros που θα κρύβουν μετατροπές σε strings, κλήσεις συναρτήσεων, κάποιους operators ή και βοηθητικά προσωρινά στιγμιότυπα.

- Τους initializers της C++11 –{}– για κλήση constructors και αρχικοποίηση μεταβλητών.