Lessons touched by this meeting according to schedule:

* 20. 16/12/2024
  + Recursive functionals (recursive operators, in the book)
  + Myhill-Shepherdson Theorem
  + First recursion theorem [§10.1, §10.2, §10.3, without proofs]
* 21. 17/12/2024
  + Immagine che contiene testo, schermata, Carattere, algebra

    Descrizione generata automaticamenteImmagine che contiene testo, schermata, Carattere

    Descrizione generata automaticamenteSecond recursion theorem [§11.1, §11.2]

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Immagine che contiene testo, Carattere, schermata, algebra

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamenteMyhill-Shepherdson Theorem:

Immagine che contiene testo, schermata, Carattere, algebra

Descrizione generata automaticamente

The First Recursion Theorem is an important result in computability theory that is used in several key ways in the exercises provided:

1. Proving the existence of fixed points for recursive functionals. The theorem states that every recursive functional has a least fixed point that is computable. This is applied in examples like showing that the Ackermann functional has the Ackermann function as its least fixed point.
2. Classifying sets as recursively enumerable or not. The theorem can be used to show that certain sets, like the set A={x | φ\_x(y)=x^2 for infinitely many y}, are likely recursively enumerable but not recursive. The proof sketch leverages the First Recursion Theorem.
3. Proving undecidability results. As a corollary, the theorem allows proving Rice's Theorem, which states that any non-trivial property of computable functions is undecidable. This in turn is used to prove the undecidability of the Halting Problem.
4. Showing the non-extensionality of certain sets. The theorem is applied to prove that the Halting Set K={x | φ\_x(x)↓} is not saturated (i.e. extensional). The diagonalization argument in the proof relies on the recursion theorem.

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamenteThe Second Recursion Theorem, also known as Kleene's Fixed Point Theorem, states that for any total computable function f : N → N, there exists a fixed point e ∈ N such that φ\_e = φ\_f(e). In other words, there is a program e that, when executed, behaves exactly like the program obtained by applying f to e.

This theorem is used in the course to:

1. Show the non-extensionality of certain sets. For example, the Halting Set K = {x | φ\_x(x)↓} can be proven to be non-extensional using the Second Recursion Theorem. The proof involves constructing a program that behaves differently when given its own index as input.
2. Demonstrate the existence of self-referential programs. The Second Recursion Theorem allows us to create programs that can access their own source code or index during execution. This is a powerful technique for constructing counterexamples and proving the limitations of computable functions.

Immagine che contiene testo, schermata, Carattere, documento

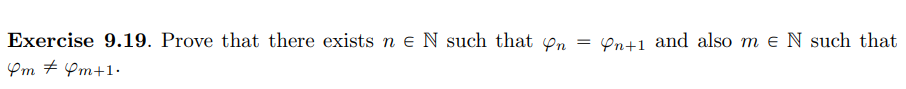
Descrizione generata automaticamenteLet’s use these proofs as exercises:

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamenteLet’s jump to some exercises:

Immagine che contiene testo, Carattere, schermata, linea

Descrizione generata automaticamenteImmagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

Here is a formal proof using the Second Recursion Theorem to show there exists n ∈ N such that φ\_n = φ\_n+1 and also m ∈ N such that φ\_m ≠ φ\_m+1:

Let s : N → N be the successor function defined by s(x) = x + 1 for all x ∈ N. The function s is total and computable.

By the Second Recursion Theorem, there exists an index e ∈ N such that

φ\_e = φ\_s(e)

Since s(e) = e + 1, we have

φ\_e = φ\_e+1

Thus, by taking n = e, we have shown there exists n ∈ N such that φ\_n = φ\_n+1.

To prove the second part, suppose for the sake of contradiction that for all m ∈ N, φ\_m = φ\_m+1. Then all computable functions would coincide, which is clearly not the case. For example, the constant functions λx.0 and λx.1 are both computable but different.

Therefore, there must exist m ∈ N such that φ\_m ≠ φ\_m+1.

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamenteThus, we have formally proved both parts of the exercise using the Second Recursion Theorem. ∎

Immagine che contiene testo, schermata, Carattere, algebra

Descrizione generata automaticamenteLet’s jump to some recursiveness exercises:

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamenteImmagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Immagine che contiene testo, Carattere, schermata, linea

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, algebra

Descrizione generata automaticamenteImmagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Immagine che contiene testo, Carattere, bianco, schermata

Descrizione generata automaticamenteImmagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente