Παρακάτω θα αναλυθούν οι συναρτήσεις που αποτελούν τον κορμό των λειτουργιών του προγράμματος και που βρίσκονται στο αρχείο processData.py. Η επεξήγηση του κυρίως προγράμματος main.py θα γίνει μέσω αυτής της διαδικασίας.

Η πρώτη συνάρτηση ονομάζεται formatData και σκοπός της είναι να μετατρέψει τα δεδομένα σε μορφή που θα επιτρέψει και θα διευκολύνει την επεξεργασία τους.

Τα δύο κύρια αρχεία δεδομένων που χρησιμοποιεί το πρόγραμμα είναι η χρονοσειρά του φορτίου του δικτύου και η χρονοσειρά της παραγώμενης ισχύος των ΦΒ, για την συγκεκριμένη περιοχή.

Τα δεδομένα για το φορτίο του δικτύου λαμβάνονται έτοιμα για την περιοχή που θα επιλεχθεί, από το αρχείο στο οποίο είναι αποθηκευμένα με όνομα data/*'περιοχή'*_gridload.csv. Ο χρήστης στην αρχή του προγράμματος ζητείται να διαλέξει μία από τις περιοχές που αναγράφονται και ανάλογα με την επιλογή του (1-5), επιλέγεται το κατάλληλο αρχείο από τον παραπάνω φάκελο.

```
print("\nSelect examination area from the list below (1-5)")
place = int(input("[1] Chania\n[2] Rethymno\n"
                  + "[3] Heraklio\n[4] Ag.Nikolaos\n[5] Moires\n"))

while place > 5 or place < 1:
    print("\nInvalid answer. Please choose one of the below:")
    place = int(input("[1] Chania\n[2] Rethymno\n"
                      + "[3] Heraklio\n[4] Ag.Nikolaos\n[5] Moires\n"))
```

Όσον αφορά τα δεδομένα για την παραγώμενη ισχύς από τα ΦΒ, αυτά κατεβαίνουν από την ιστοσελίδα του PV-GIS μέσω του API του. Το αίτημα για τα δεδομένα γίνεται μέσω της εντολής curl και τα δεδομένα αυτά αποθηκεύονται με τη σειρά τους στο αρχείο data/pv_production.csv. Για να γίνει το αίτημα αυτό στην ιστοσελίδα, θα πρέπει να δωθούν κάποιες παράμετροι, μέσα στις οποίες είναι και οι συντεταγμένες της περιοχής. Συνεπώς ανάλογα με την επιλογή του χρήστη, οι συντεταγμένες αποθηκεύονται σε δύο μεταβλητές, lat και lon. Αυτές αντλούνται από ένα dictionary, το οποίο ορίζεται στην αρχή του προγράμματος για να είναι εφικτή η εύκολη τροποποίησή του ανάλογα με τις ανάγκες του χρήστη. Για τον ίδιο λόγο ορίζονται στην αρχή του προγράμματος και οι υπόλοιπες μεταβλητές που χρειάζεται η ιστοσελίδα, αλλά και όλο το πρόγραμμα.

```
places = {
    1: (35.512, 24.012),
    2: (35.364, 24.471),
    3: (35.343, 25.153),
    4: (35.185, 25.706),
    5: (35.050, 24.877)
}
```

```
lat = str(places[place][0])
lon = str(places[place][1])
```

Η συνάρτηση formatData δέχεται ως ορίσματα τις δύο χρονοσειρές και τις μετατρέπει σε έναν πίνακα που η κάθε του σειρά είναι μία μέρα του χρόνου και κάθε στήλη του είναι η αντίστοιχη ώρα της ημέρας, κατά την οποία πάρθηκε η μέτρηση. Στο τέλος, η συνάρτηση επιστρέφει δύο πίνακες οι οποίοι έχουν 365 γραμμές και 24 στήλες.

Για την μοντελοποίηση του συστήματος αποθήκευσης θα χρειαστεί να δημιουργηθεί ένα αντικείμενο μπαταρίας, το οποίο θα μπορεί να φορτίζει και να ξεφορτίζει, να περιέχει όλα τα λειτουργικά χαρακτηριστικά της μπαταρίας και να μπορεί να αυξάνεται η χωρητικότητά της, με το να προστίθεται σε αυτήν και άλλες ίδιες μπαταρίες, σαν να υπήρχε ένα σύμπλεγμα

μπαταριών. Στην αρχή του προγράμματος δίνεται η δυνατότητα στον χρήστη να επιλέξει το είδος της μπαταρίας, από την οποία θα αποτελείται το σύστημα.

```
print("\nChoose battery type:")
type = int(input("[1] Lead-Carbon (300,000.00/MWh)\n"
                 + "[2] Lithium-Ion (500,000.00/MWh)\n"))

while type > 2 or type < 1:
    print("\nInvalid answer. Please choose one of the below:")
    type = int(input("[1] Lead-Carbon\n"
                     + "[2] Lithium-Ion\n"))

if type == 1:
    bat_type = path + "/thesis/data/lead_carbon.json"
else:
    bat_type = path + "/thesis/data/lithium_ion.json"
```

Η επιλογή του χρήστη μετατρέπεται σε όνομα αρχείου json. Με βάση αυτό το αρχείο δημιουργείται η αρχική αναπάρασταση μίας μπαταρίας. Αν το αρχείο δεν βρίσκεται στην σωστή μορφή ή δεν υπάρχει καθόλου, το πρόγραμμα τερματίζει με το να ενημερώνει για το είδος του σφάλματος.

```
try:
    bat = Battery.from_json(bat_type)
except Exception as err:
    print("\nFailed to instantiate battery object from json file...\n")
    sys.exit(err)
```

Οι υπολογισμοί που ακολουθούν απο εδώ και πέρα, εφαρμόζονται για κάθε σειρά των πινάκων φορτίου δικτύου και παραγωγής ΦΒ, δηλαδή για κάθε ημέρα του χρόνου. Στη συνέχεια αυξάνονται οι μπαταρίες κατά μία και ξανατρέχει το πρόγραμμα από την αρχή. Ουσιαστικά υπάρχουν δύο βρόγχοι for ο ένας μέσα στον άλλο. Το πρόγραμμα τερματίζει όταν ένα από τα παρακάτω σενάρια επιτευχθεί.

1. Όταν όλη η παραγόμενη ενέργεια από τις ΑΠΕ, αξιοποιείται.
2. Όταν οι ΑΠΕ μαζί με τις μπαταρίες τροφοδοτούν το 100% του δίκτυου.
3. Όταν το κόστος του συστήματος αποθήκευσης, φτάσει το μέγιστο αποδεκτό από τον χρήστη.

Για να επιτευχθεί η τρίτη συνθήκη, δίνεται η δυνατότητα στον χρήστη να εισάγει ένα μέγιστο κόστος που μπορεί να διαθέσει. Ο χρήστης μπορεί να εισάγει τον αριθμό 0, αν δεν θέλει να υπάρχει κάποιος οικονομικός περιορισμός.

Μετά την formatData εκτελείται η συνάρτηση wastedEnergy. Αυτή δέχεται για ορίσματα δύο σειρές αριθμών, στην περίπτωσή μας τις μετρήσεις μίας ημέρας για το φορτίο του δικτύου και της παραγώμενης ισχύς των ΦΒ και στη συνέχεια υπολογίζει πόση ενέργεια από τα ΦΒ δεν αξιοποιείται μέσα στην ημέρα. Στο τέλος ελέγχει αν η μπαταρία έχει χώρο να αποθηκεύσει αυτήν την ενέργεια και αν μπορεί, το κάνει.