Defensive Programming

We have discussed several situations where the software has to deal with user input and we improved the code to handle unexpected input.

The first example for defensive programming is the method to create new products.
Location: task3\src\core\Controller.java

In our project we have three different product types. We used a switch statement to generate the right Product. To handle with false input we have a default case which throws an illegalargumentexception. It is granted that only the right product is created.

```
public void createProduct(int productType, String productName, String description) {

  Product product = null;

  switch (productType) {
    case Product.PRODUCT_TYPE_CLOUD:
      product = new ProductCloud(productName);
      break;
    case Product.PRODUCT_TYPE_EMAIL:
      product = new ProductEmail(productName);
      break;
    case Product.PRODUCT_TYPE_VPN:
      product = new ProductVPN(productName);
      break;
    default:
      throw new IllegalArgumentException("Product type not found");
  }

  if (product != null) {
    product.setProductDescription(description);
    addEmptyProduct(product);
  } else {
    throw new NullPointerException("Product could not be added");
  }

}
```

The second example of defensive programming is the method to add an empty Product.
Location: task3\src\core\Controller.java

Before an empty product is added, it is checked if this product already exists. When the product already exists we throw an illegalargumentexception.

```
  private void addEmptyProduct(Product product) {

  if (!emptyProducts.contains(product)) {
    emptyProducts.add(product);

  } else {
    throw new IllegalArgumentException("The same product already exists");
  }

}
```