<https://www.nal.usda.gov/human-nutrition-and-food-safety/dri-calculator>

https://www.fda.gov/food/food-labeling-nutrition/nutrition-information-cooked-seafood-purchased-raw

Harris-Benedict Caloric Calculator

**Caloric need calculation**

* Inputs:
  + Gender
  + Age
  + Height
  + Weight
  + Activity level
* Output
  + BMR:
    - Men: BMR = 66.5 + (13.75 × weight [kg]) + (5.003 × height [cm]) – (6.775 × age [years])
    - Women: BMR = 655.1 + (9.563 × weight [kg]) + (1.850 × height [cm]) – (4.676 × age [years])
  + Activity level:
    - Sedentary (little or no exercise): AMR = BMR x 1.2
    - Lightly active (exercise 1–3 days/week): AMR = BMR x 1.375
    - Moderately active (exercise 3–5 days/week): AMR = BMR x 1.55
    - Active (exercise 6–7 days/week): AMR = BMR x 1.725
    - Very active (hard exercise 6–7 days/week): AMR = BMR x 1.9

Conclusion: the number of calories your body needs for a day.

**Menu**

* Input:
  + Food avoidance (allergies/religious/obligations/out-of-access): find and select.
  + Preference:
    - No preference (0)
    - Asian and Pacific (1)
    - European and North America (2)
    - Mediterranean & Hispanic (3)
    - Vegan (4)
  + Desired progress: goal:
    - Gain (1)
    - Lose (-1)
    - weight maintenance (0)
  + Weight gain or lose: amount: x (kg).
  + Duration: weeks
    - Default: 8 weeks
  + Number of main meals: *main int*
    - 2 or 3
    - Default: 3
    - main = 2 => lunch 6, dinner 4
    - main = 3 => breakfast 3.5, lunch 3.5, dinner 3
  + Number of side meals: *side int. 1 side meal = 100 Kcal*
    - 0 or 1 or 2
    - Default: 1
  + Number of exercise sessions: *session int*
    - 1 or 2 or 3
    - Default: 1
* Output:
  + Calculation:
    - Real caloric need:
      * Weight maintenance: total = AMR
      * Weight gain/lost:
        + Calories gain or lose: x\*7716 (Kcal)
        + Calories gain or lose: per day:
    - => caloMass = AMR + or –
  + Plate proportion based on preference:
  + STANDARD

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Preference | No Preference (0) | Asian and Pacific (1) | European and North America (2) | Mediterranean & Hispanic  (3) | Vegan  (4) |
| Carbohydrate | 50% | 60% | 40% | 47% | 60% |
| Fiber | Carbohydrate/11 | Carbohydrate/11.2 | Carbohydrate/13 | Carbohydrate/12 | Carbohydrate/9 |
| Protein | 0.8\*weight  (check 25%) | 0.8\*weight  (check 20%) | 0.9\*weight  (check 30%) | 0.85\*weight  (check 25%) | 0.9\*weight  (check 22%) |
| Fat | 25% | 20% | 30% | 28% | 18% |
| Water | 2250 g | 2250 g | 2500 g | 2500 g | 2250 g |

###TẤT CẢ CÁC CHỈ SỐ TRÊN NÊN ĐƯỢC NHÂN VỚI \*0.9 TRONG CÔNG THỨC MỖI BỮA ĂN ĐỂ TRỪ HAO VƯỢT MỨC QUY ĐỊNH. NẾU CHỈ SỐ NÀO THIẾU THÌ SẼ ĐƯỢC GỢI Ý TRONG PHẦN SẢN PHẨM.

###Các bữa phụ được tính dựa trên mức calo đơn thuần: amount = SIDE\_MEAL\_SIZE\*size/caloricintake

###Các bữa chính:

* Carbohydrate từ starch: 0.8\*lượng carbohydrate tiêu chuẩn
* Fiber từ vegetables and fruit, in which vegetables occupy 70% of fiber need and fruit takes the rest.
* Protein từ fish/seafood/egg/meat/poultry. Trong đó, bữa trưa và tối có 2 nguồn protein, mỗi nguồn chứa 50% lượng protein cần.
* Fat từ nguồn protein + phần còn thiếu từ dairies&dessert
* Nước lấy từ drink: 0.25\*lượng nước tiêu chuẩn cho mỗi main meal

###Ví dụ:

* A person, weighed 60 kg, demands 2200Kcal for his life sustainability. He decided on a no-preference diet with 3 main meals and 2 side meals. Design an appropriate menu.
* SIDE MEALS:
  + Brunch:
    - Caloric intake: 100 Kcal
    - Random food selected:
      * Peanut: amount = 100\*146/832 = 17.5 g
  + Supper:
    - Caloric intake: 100 Kcal
    - Random food selected:
      * Kefir: amount = 100\*244/127 = 192.1 g
* MAIN MEALS: total caloric need = 2200 – 2\*100 = 2000 Kcal
  + Breakfast:
    - Caloric intake = 2000\*0.35 = 700 Kcal
    - Carbohydrate = 700\*50%\*0.8 = 280 Kcal = 70 g
      * Random food selected:
        + Tacos shell: amount = = 58.95 g
    - Fiber = carbohydrate /11 = 280/11 = 25.45 Kcal
      * Random food selected:
        + Kale: amount = (0.7\*25.45)\*21/7.4 = 50.6 g
        + Cherry: amount = (0.3\*25.45)\*150/95 = 12.1 g
    - Protein = weight\*0.8\*0.35 = 60\*0.8\*0.35 = 16.8 g
      * Random food selected: pork : amount = 16.8 g
    - Fat so far =

ALGORITHM:

1. Create a new user with inputs from the form above and insert user to the database:
   * DEFAULTS:
     + private int final DEFAULT\_PREFERENCE = 0
     + private int final DEFAULT\_AMOUNT = 2
     + private int final DEFAULT\_WEEK = 8
     + private int final DEFAULT\_MAIN = 3
     + private int final DEFAULT\_SIDE = 1
     + private int final DEFAULT\_SESSION = 2
     + private int final DEFAULT\_RANK = 1
   * private int numUser = the number of registered users
   * private String userID = String.format(“U%05d”,++numUser);
   * private String username = Utils.toTitleCase(request.getParameter(“username”));
   * private Date dob = Utils.parseDate(request.getParameter(“dob”)); //try catch
   * private String phone = request.getParameter(“phone”);
   * private String email = request.getParameter(“email”);
   * private String password = request.getParameter(“password”);
   * private double weight = Double.valueOf(request.getParameter(“weight”));
   * private double height = Double.valueOf(request.getParameter(“height”));
   * private int gender = Integer.valueOf(request.getParameter(“gender”));
   * private int activity = Integer.valueOf(request.getParameter(“activity”));
   * private int preference = (request.getParameter(“preference”) == null)? DEFAULT\_PREFERENCE:Integer.valueOf(request.getParameter(“preference”));
   * private int goal = Integer.valueOf(request.getParameter(“goal”));
   * private double amount;
     + if (goal == 0) amount = 0; else if (request.getParameter(“amount”)) == null) amount = DEFAULT\_AMOUNT else amount = goal\*Integer.valueOf(request.getParameter(“amount”));
   * private int duration = (request.getParameter(“duration”) == null)? DEFAULT\_DURATION: Integer.valueOf(request.getParameter(“duration”));
   * private int main = (request.getParameter(“main”) == null)? DEFAULT\_MAIN: Integer.valueOf(request.getParameter(“main”));
   * private int side = (request.getParameter(“side”) == null)? DEFAULT\_SIDE: Integer.valueOf(request.getParameter(“side”));
   * private int session = (request.getParameter(“session”) == null)? DEFAULT\_SESSION: Integer.valueOf(request.getParameter(“session”));
   * private int rank = DEFAULT\_RANK;
   * private Date createdate;
     + DateTimeFormatter dtf = DateTimeFormatter.ofPattern("MM/dd/yyyy");
     + LocalDateTime now = LocalDateTime.now();
     + createdate = dtf.format(now);
   * User o = new User(userID);
   * Setters
2. Insert table Day:

* private int numDay = the number of days in the table Day
* private String dayID;
* private int index = 0;
* for(int i = 0; i < duration\*7; i++){

dayID = String.format(“DAY%06d”,++num);

DayManager.add(new Day(dayID, userID, ++index));

}

1. Prepare the Food dataset that fits user’s preference and avoidance:
2. Create meals:

* CONSTANTS:
  + private final int BREAKFAST\_INDEX = 1
  + private final int LUNCH\_INDEX = 2
  + private final int DINNER\_INDEX = 3
  + private final int BRUNCH\_INDEX = 4
  + private final int SNACK\_INDEX = 5
  + private final int SIDE\_MEAL\_SIZE = 100
  + private final double LUNCH\_2 = 0.5
  + private final double DINNER\_2 = 0.5
  + private final double BREAKFAST\_3 = 0.35
  + private final double LUNCH\_3 = 0.35
  + private final double DINNER\_3 = 0.3
* private HashMap<Integer, double> mealIndex = new HashMap();
* double caloRemainder = 0;
* caloRemainder = caloMass – side\*SIDE\_MEAL\_SIZE;
* //Side meals
* if (side == 2){

mealIndex.put(4, SIDE\_MEAL\_SIZE);

mealIndex.add(5, SIDE\_MEAL\_SIZE);

} else if (side == 1){

if (main == 2)

mealIndex.put(4, SIDE\_MEAL\_SIZE);

else

mealIndex.put(5, SIDE\_MEAL\_SIZE);

}

//Main meals

if (main == 2){

mealIndex.put(2, LUNCH\_2\*caloRemainder);

mealIndex.put(3, DINNER\_2\*caloRemainder);

} else{

mealIndex.put(1, BREAKFAST\_3\*caloRemainder);

mealIndex.put(2, LUNCH\_3\*caloRemainder);

mealIndex.put(3, DINNER\_3\*caloRemainder);

}

* private numMeal = the number of current meals in the table Meal
* for(int i = 0; i < numDay; i++){

for(Integer j : mealIndex.keySet()){

String mealID = String.format(“MEAL%05d”, ++numMeal);

int time = j;

double calosize = mealIndex.get(j);

MealManager.add(new Meal(mealID, userID, DayManager.get(i), time, calorsize));

}

}

1. Create food details

* FOOD DETAIL GENERATION
  + Constraints:
    - Breakfast types: vegetables + meat/poultry + starch + fruit
    - Lunch + Dinner: vegetables + fish/seafood/egg + meat/poultry + starch + fruit/dairies&dessert + drink
    - Side meal types: fruit/ nuts/ legumes/ dairies&dessert
  + random foodID by index.
  + double ratio = SIDE\_MEAL\_SIZE/caloricintake
  + MealList mealList = MealManager.getMealList();
  + private double amount;
  + double ratio;
  + int totalCal
  + double carbohydrate
  + double fiber
  + double protein
  + double fat
  + double water
  + for(int j = 0; j < mealList.size(); j++){

if(mealList.get(j).getTime() >= 4){ //side meals

Get a random food from side meal dataset.

ratio = mealList.get(j).getCalosize()/food.getCaloricintake();

amount = ratio\*food.getSize();

FoodDetailManager.add(new FoodDetail());

//Update the accumulative amounts

carbohydrate += ratio\*food.getCarbohydrate();

fiber += ratio\*food.getFiber();

protein += ratio\*food.getProtein();

fat += ratio\*food.getFat();

water += ratio\*food.getWater();

}else if (mealList.get(j).getTime() == 1){ /breakfast

Get a random food from side meal dataset.

}

}