Frederick Zuberer ([fzuberer@kent.edu](mailto:fzuberer@kent.edu))

Dan Maher ([dmaher2@kent.edu](mailto:dmaher2@kent.edu))

Anthony Telerico ([ateleric@kent.edu](mailto:ateleric@kent.edu))  
Winson Li ([wli21@kent.edu](mailto:wli21@kent.edu))

Step 1: HomeScreen //User Opens Application

* Display Home Screen
  + Version Select
  + Open Term Definition
* User Selects Version
  + Begin Calculator, move to step 2

Step 2: attacks\_Func //User Inputs the number of attacks being made, receives no output

* Display Prompt “Enter number of attacks being made”
* User inputs number of attacks
  + Integer type, 0 < #ofAttacks < 500
  + #ofAttacks saved
  + #ofAttacks passed to Step 3

Step 3: ToHit\_Func //User inputs Ballistic Skill/Weapons Skill, modifiers, and re-rolls, receives statistical average number of hits expressed as “Average hits”

* Display Prompt “Enter Units Ballistic skill or Weapon Skill\_+”
* Display definition of Ballistic Skill and Weapon Skill
  + BWSkill, Integer 2<= X <=6
    - User inputs incorrect BWSkill, return error message
* Display Prompt “Enter hit roll modifier \_, note can only be +1 or -1. If none enter 0”
  + Take BWSkill, add hit roll modifier-> new value of BWSkill
  + Convert into a double, so BWSkill/6-> 1-BWSkill -> BWSkillMod
  + If BWSkill = 6, save BWSkill mod with .1666
* Display Definition of a Hit Roll Modifier
* Display Prompt “Do you have Re-roll 1’s”
  + Boolean, if T apply Re-Roll 1 mod to BWSkillMod, if F do nothing
* Display Prompt “Do you have Re-roll all?”
  + Boolean, if T apply Re-Roll all mod to BWSkillMod, if F do nothing
* Display Definition of “Re-Rolls”
* Take #ofAttacks, pass value to Hits
* Multiply hits by final BWSkill value
* Output hits as “Average hits”

Step 4: ToWound\_Func //Users inputs Strength of weapons being used vs Toughness of target, receives “Average Wounds” as output

* Display Prompt “Enter Weapons Strength”
* Display Definition of Strength
* Display Prompt “Enter Targets Toughness”
* Display Definition of Toughness
  + Store Strength as integer variable Str
  + Store Toughness as integer variable Tough
  + Pass to StrVsTough\_Func //A sub function within ToWound\_Func
    - StrVsTough\_Func takes Str and Tough, compares, returns WoundMod
* Display Prompt “Enter wound roll modifier \_, note can only be +1 or -1. If none enter 0”
  + Take WoundMod, add WoundRollMod ->WoundMod // new value
* Display Definition of a Hit Roll Modifier
* Display Prompt “Do you have Re-roll 1’s, if you have re-roll all input F for this prompt”
  + Boolean, if T apply Re-Roll 1 mod to WoundMod, if F do nothing
* Display Prompt “Do you have Re-roll all?”
  + Boolean, if T apply Re-Roll all mod to WoundMod, if F do nothing
* Display Definition of “Re-Rolls”
* Take Hits, pass value to Wounds
* Multiply Wounds by final WoundMod
* Output wounds as “Average Wounds”

Step 5: Damage\_Func // User inputs the various save characteristics and receives “Average damage” as output

* Display Prompt “Enter Your Weapons Damage, if random input as DX+ any additions. Examples are Melta weapons in Melta Range, D6+2”
  + Some weapons have random damage, DamCal\_Func returns AvDam
* Display Prompt “Enter Targets Armor Save”
  + Take the save as an integer, ArmSav
* Display Prompt “Enter Your weapons AP”
  + Note, 0 >= X >= -6
  + Take the AP as an integer
  + Multiply AP by -1 -> ApMod
* Display Prompt “Is Target in Cover?”
  + Boolean, if T -1 to ApMod, if F do nothing
* Final Save Calculation
  + ApMod added to Save
* Display Prompt “Does the Target have an Invulnerable Save, if no enter 0”
  + Take Invulnerable Save, store as Invuln.
  + If Invuln = 0, pass ArmSav to FinSav
  + if Invuln > 0 and Invuln < ArmSav, pass Invul FinSav
  + If Invuln > ArmSav pass ArmSav to FinSav
* Convert FinSav to SavMod
  + FinSav/6
  + If FinSav/6 greater than 1, floor to 1, save FinSav as 1
  + If FinSav/6 < 1, save FinSav with new value
  + AvDam multiplied by FinSav
* Display “Current Average Damage done” //Display AvDam
* Display Prompt “Enter the Targets Feel no Pain save?”
  + Take integer, 2<= X <=6, Save as FNPSav
  + Convert to Modifier, FNPSav/6 -> 1-FNPSav -> FNPMod
  + If FNPSav = 6, save with .1666
  + Multiply AvDam by FNPMod, save AvDam with new value

Step 6: output\_Display

* Display “On average you will deal X damage to the target” // Display AvDam