Problem Statement: This program will ask for certain number of dice sides, then the number of those dice that are being rolled, adding the results until the number of dice has been met. In the meantime it will also roll a d20 to simulate an actual attack roll. If the result is a 1 it will display that the user died. If it is a 20, it will double the damage to be calculated. Otherwise it will determine the roll plus modifier to determine if the attack was successful in dealing damage. Then it will ask for any damage and attack modifiers to apply to the roll. It will then ask for the armor class of the enemy and if the result of the roll plus the attack modifier is higher, then it will display the damage, otherwise just display that it was a missed attack.

Nouns: program, dice sides, dice, results, number of dice, damage, attack modifiers, roll, it, armor class, enemy

Verbs: will ask, rolled, adding, met, apply, is higher, displays

Inputs: Number of sides, number of dice, attack modifier, damage modifier, armor class

Outputs: result of d20 roll, result of total attack roll, sum of damage dice

Processes: d20 roll =random 1-20; d20 roll + modifier >= armor class, rolls damage dice (number sides = random 1-numbersides; damage roll = numbersides \* (numberdice \* numbersides) + modifier. If d20 = 20, doubles damage, if d20 = 1, displays “you are dead”. If d20 + modifier < armor class, display ‘result of d20 roll + modifier + “was a miss against an enemy with an armor class of “ + armorclass’.

PSEUDOCODE:

BEGIN

Function diceRoll()

Ask for numberSides, numberDice, hitMod, damMod, armorClass

Initialize count

Initialize sum

Roll d20

IF d20 = 1 THEN

Write “you are dead”

ENDIF

WHILE count < numberDice THEN

Roll numberSides

Add sum

Increment count

ENDWHILE

ENDIF

IF d20 != 1 & d20 !=20 THEN

IF d20 + hitMod >= armorClass THEN

Write sum + damMod

ELSEIF d20 + hitMod < armorClass THEN

Write d20 + hitMod “missed”

ENDIF

ENDIF

IF d20 = 20 THEN

Write “crit!”

Write sum\*2 + damMod \*2

ENDIF

ENDFUNCTION

TESTPLANS:

Inputs: numberSides = 6, numberDice = 1

Output: number between 1-20

Number between 1- 6

Outcome: 12, 4; 9,4; 1,2; 7,2;