Assignment 4 Design
Tatyana Vlaskin

1. First modification that I need to make is to make my parent class.
2. Two functions are used by 4 child classes: int attack() and int defense().
3. These functions are slightly different in each child class, but have the same number of parameters. I will redefine the function in each child class and make these functions virtual. This will allow the base point and use the most-derived version of the function that it finds.
4. So the parents class will look somewhat like this. Please note that I'll make attack() and defense() functions pure virtual, by setting them equal to zero:

```
  class CHARECTOR{
protected:
  string name;
  int atackNumberOfDiceRoles;-- will get rid of this variable, see below for
explanation
  int atackDiceSides; ;-- will get rid of this variable, see below for explanation
  int armor;
  int defenceNumberOfDiceRoles; ;-- will get rid of this variable, see below for
explanation
  int defenceDiceSides; ;-- will get rid of this variable, see below for explanation
  int damage;
  int strength;
  int ststore;// this variable will be used to calculate the % of the strenght points
left
public:
int getarmor(){return(armor);}
int getstrength(){return(strength);}
void setstrength(int strength2){    strength=strength2;    }
virtual int attack() =0;{
virtual int deffence()=0;{
int getstnstore(){return(ststore);}
int getdamage(){return(damage);}
string getname(){return(name);}
```

5. Next change will be to modify my child classes.
6. Currently my child classes have on constructor in them.
7. Each class will get 2 virtual functions. One for attack and one for defense.
8. My current attack function in the parent class look like this:

```
      int attack(){
        int sum=0;
        int randomAttackPoints;// variable to keep track of atack points
        cout << endl << name << " - attack ";// will display a message who is
attacking
```

```
        for(int i=0; i<atackNumberOfDiceRoles; i++){// the dice will be rolled as
    many times as indicated for each character
            randomAttackPoints = rand()%atackDiceSides + 1; // the values of
    attack will be between 1 and sides of atack points
            cout << randomAttackPoints << " ";// message that will display how many
    atack points are generated
            sum+=randomAttackPoints;
        }
        cout<<"Sum : " << sum << endl;// message that will indicate total attack
    points
        return(sum);
      }
```

9. AtackDiceSides and **atackNumberOfDiceRoles variables will be removed from the base class and in the actual attack functions they will be replaced with the appropriate values in each class during the function redefinition.**
10. **Similarly** int defenceNumberOfDiceRoles and int defenceDiceSides variable will be deleted from the base class and in the defense functions they will be replaced with the values corresponding to each character.
11. As the result of the just described change, the constructor of the child classes, needs to be changed as well. AtackDiceSides, **atackNumberOfDiceRoles,** defenceNumberOfDiceRoles and int defenceDiceSides variable will be removed.
12. New child classes will look somewhat like that

```
///CLASS GOBLIN
class Goblin:public CHARECTOR{
public:
Goblin(){name="Goblin"; armor=3; strength=ststore=8; }
virtual int attack(){
for(int i=0; i< 2 ; i++){// need to roll attack dice x2
        randomAttackPoints = rand()%6 + 1; // 6 sides on the attack dice
}
virtual int deffence(){
for(int i=0; i< 1 ; i++){// need to roll deffene dice x1
        randomDeffencePoints = rand()%6 + 1; // 6 sides on the defence dice
}

///CLASS BARBARIAN
class Barbarian:public CHARECTOR{
public:
  Barbarian(){name="Barbarian";armor=0;strength=ststore=12}
virtual int attack(){
for(int i=0; i< 2 ; i++){// need to roll attack dice x2
        randomAttackPoints = rand()%6 + 1; // 6 sides on the attack dice
}
```

```
virtual int deffence(){
for(int i=0; i< 2 ; i++){// need to roll deffene dice x2
        randomDeffencePoints = rand()%6 + 1; // 6 sides on the defense dice
}
///CLASS REPTILEPEOPLE
class ReptilePeople:public CHARECTOR{
public:
  ReptilePeople(){name="ReptilePeople";armor=7;strength=ststore=18; }
virtual int attack(){
for(int i=0; i< 3 ; i++){// need to roll attack dice x3
        randomAttackPoints = rand()%6 + 1; // 6 sides on the attack dice
}
virtual int deffence(){
for(int i=0; i< 1 ; i++){// need to roll deffene dice x1
        randomDeffencePoints = rand()%6 + 1; // 6 sides on the defence dice
}
///CLASS BLUEMEN
class BlueMen:public CHARECTOR{
public:
  BlueMen(){name="BlueMen";armor=3;strength=ststore=12;}
virtual int attack(){
for(int i=0; i< 2 ; i++){// need to roll attack dice x2
        randomAttackPoints = rand()%10 + 1; // 10 sides on the attack dice
}
virtual int deffence(){
for(int i=0; i< 3 ; i++){// need to roll deffene dice x3
        randomDeffencePoints = rand()%6 + 1; // 6 sides on the defence dice
}
```

13. Next step will to be create one new character class. Ill create a Mega Man. He will be derived from the Barbarian class. Description of the Mega Men from the previous assignment is provided below.

Mega Man

Any damage that would normally be applied to strength gets diverted to an energy store. The energy store can be used to make an enhanced attack. Any damage received in excess of the store is applied to strength as usual.

14. After reading the description it become obvious that we need to introduce new variable, lets call it EnergyStore. Additionally, let's introduce one more attack function. This attack function will use enchanted weapon.

```
///CLASS MEGA MEN
```

```
class MegaMen:public Barbarian{
public:
//I'll randomly choose armor, strength and EnergyStore of the Mega men
//also attack function will be inherited from the barbarian and will not be redefined
  MegaMen (){name=" Mega Men ";armor=10;strength=ststore=20; EnergyStore
=10;}
virtual int deffence(){
for(int i=0; i< 2 ; i++){// need to roll deffene dice x2
        randomDeffencePoints = rand()%6 + 1; // 6 sides on the defense dice
}
```

The main difference between this class and any other character classes will be that the impact on health after attack is different.
Any damage that would normally be applied to strength gets diverted to an energy store. The energy store can be used to make an enhanced attack. Any damage received in excess of the store is applied to strength as usual.
As a result of this when the strength of the MegaMen is calculated after the battle, it will be done somewhat like this:

```
int netAttackPoints=player.attack()-MegaMen.deffence();    netAttackPoints=
netAttackPoints - MegaMen.getarmor();
   if(netAttackPoints<0){
      netAttackPoints=0;
   }
MegaMen NewEnergy= MegaMen.getstrength()-netAttackPoints;
   if(MegaMen NewEnergy <0){
         MegaMenNewEnergy = absolute value of (MegaMenNewEnergy)          Meg

  }
```

And finally, we need to reset strength and energy store:

```
   MegaMen.setstrength(MegaMenNewStrength);
   MegaMen.setstrength(MegaMenNewEnergy);
```

15. Because of the MegaMen, it becomes obvious that we need to create a way to use enchanted weapon. In order to user enchanted weapon, the character needs to have energy store points- see blue man for description. As a result of this, Ill add a variable to the base class and call it Energy. Each character will get some energy points that they will be able to use to fight with enchanted weapon. Once, the character runs out of the energy points, they will not be able to use enchanted weapon anymore and will be forced to use regular attack function. MegaMen is the only character that will be able to use energy for defense as well. All other characters will be able to use energy only during the attack. Each time an enchanted weapon is used, a character loses 1 energy point.

a. **///CLASS GOBLIN - 3 energy points**
b. **///CLASS BARBARIAN -4 energy points**
c. **///CLASS REPTILEPEOPLE-5 energy points**
d. **///CLASS BLUEMEN-6 energy points**
e. **///CLASS MEGA MEN-10 energy points**

16. Ok back to the enchanted weapon. During the combat, a character will be asked if they want to use regular weapon or enchanted weapon. If they have energy points they will be allowed to use enchanted weapon that will impose x2 points of attack. Same attack function will be used. However, whatever value is produced by the attack function will be multiplied by 2 and that will be the resultant attack points. Attack points will be subtracted from the strength points. Please note only Mega Man will be able to use energy during defense.
17. Next, I need to introduce venom. According to the description that was provided in the assignment, we need to do something like this:
    A single bite, sting, spit whatever inserts venom in the target. For a speci fied number of turns applies constant or degrading damage to the target's strength. As it's not physical energy
    Mega Man cannot divert the damage to the energy store but is taken aga inst strength.
    This does not make such sense to me. What I will do is that if the character is attacked with the venom, their defense and attack points will be reduced by certain %.
    I think to make testing easier, Ill give each character only 1 bottle of venom. The will be an option where each character is asked if they want to attack with venom. No damage to the health of the opponent will be done, but their defense and attack points will be reduced by certain % depending who imposes a venomous damage for the next 2 rounds. Thus, there will be another variable added into the classes, lets call it VenomeDamage.
    a. **///CLASS GOBLIN - 5% imposed damage**
    b. **///CLASS BARBARIAN -5% imposed damage**
    c. **///CLASS REPTILEPEOPLE-20% imposed damage**
    d. **///CLASS BLUEMEN-7% imposed damage**
    e. **///Mega Men-15% imposed damage**
18. One more think that will be added to the program is spinach. To make testing simple, Ill let the character each the spinach before the combat. If the character decides to eat spinach, their attack points will quadruple, and maybe finally, my goblin will be able to win. It was very weak character in the last battle game. As a result of the spinach, each class will have another attack function that will reflect spinach.
19. Another change that I need to make to my program is make sure that base class is never called, thus, I need to change my switch statements:
    I'll delete the following 2 objects: CHARECTOR player,enemy;
    Goblin goblin1;

```
Barbarian barbarian1;
ReptilePeople reptile1;
BlueMen blueMen1;
MegaMen megaMen1;

CRectangle rectangle;
CTriangle triangle;

CHARECTOR * ptr_ goblin  = & goblin1;
CHARECTOR * ptr_ barbarian = & barbarian1;
CHARECTOR * ptr_ reptile = & reptile1;
CHARECTOR * ptr_ blueMen = & blueMen1;
CHARECTOR * ptr_ megaMen = & megaMen1;

 switch(result){
    case 1:
         CHARECTOR * ptr_ goblin  = & goblin1;
       break;
    case 2:
       CHARECTOR * ptr_ barbarian = & barbarian1;
        break;
etc.
```

As a result of this when I'll try to get attack or defense functions, I need to do something like this:

```
ptr_ goblin  ->defence ()
```

Testing:

In summary:

The following battle combinations need to be tested with all commendations of characters. Lest use barbarian and goblin as an example.

1. None of the characters use spinach, none of the characters use enchanted weapons and none of the characters use venom
2. Goblin uses spinach, barbarian does not use spinach, neither goblin nor barbarian use enchanter weapon or venom
3. Barbarian uses spinach, goblin does not use spinach, neither goblin nor barbarian use enchanter weapon or venom
4. None of the characters use spinach, barbarian uses enchanted weapon until runs out of energy point, goblin uses only regular attack and none of the characters use venome
5. None of the characters use spinach, goblin user enchanted weapon until runs out of energy points, barbarian does not use enchanted weapon, none of the characres us venom

6. None of the characters use spinach, both characters use enchanted weapons until they run out of energy , neither goblin nor barbarian use enchanter weapon or venom
7. None of the characters use spinach, both characters use enchanted weapon randomly – this is complicated.
8. Do tests indicted in 4-7, when both characters use spinach
9. Do tests indicated in 4-7 when only barbarian uses venom
10. Do tests indicated in 4-7 when only goblin uses venom
11. Do tests indicated in 4-7 when both goblin and barbarian use venom
12. Do step 8 when only barbarian uses venome
13. Do step 8 when only goblin uses venom
14. Do step 8 when both goblin and barbarian use venom
15. Do the above mentioned tests for different character combinations.

I have to do similar testing that were done in the last battle game, so I am copying and pasting the chart from the last game here.

Whats are we testing	How we are testing	What is expected	What is the output	PASS/FAIL
Make sure that radom number of attack points is generated each time	cout << randomAttackPoints << " sum+=randomAttackPoints;	Each time there is a different attack points	Each time there is a different attack points	PASS
Make sure Goblin rolls the dice x2 for attack	There is a for loop with the : cout << randomAttackPoints << " sum+=randomAttackPoints;	2 values for attack should be displayed	2 values for attack should be displayed	PASS
Make sure Goblin gets values from 1-6 for each dice roll	There is a for loop with the : cout << randomAttackPoints << " sum+=randomAttackPoints;	Values of attack1 and attack 2 between **1-6**	Values of attack1 and attack 2 between **1-6**	

Make sure Goblin attack1 and attach2 value add up and assigned to total attack points	There is a for loop with the : cout << randomAttackPoints << " sum+=randomAttackPoints;	Attack will be equal to sum of 2 dice roles	Attack will be equal to sum of 2 dice roles	PASS
Do similar tests described for Goblin for Barbarian	See Goblin	See goblin	See goblin	PASS
Do similar tests described for Goblin for Reptile	Similar to Goblin, the difference is that, there have to be 3 rolls of dice and dice values are from 1 to 3. There is a for loop with the : for(int i=0; i<atackNumberOfDiceRoles; i++){ cout << randomAttackPoints << " sum+=randomAttackPoints	Make sure that are attack1, attack2 and attack3 have value and they are in the range **1-3**	Make sure that are attack1, attack2 and attack3 have value and they are in the range **1-3**	PASS
Do similar tests described for Goblin for Blue Man	Similar to Goblin, the different is that, there have to be 2 rolls of dice and dice values are from 1-10.	Make sure that are value for attack 1 and attack2 and values are between **1-10**	Make sure that are value for attack 1 and attack2 and values are between **1-10**	PASS
Make sure Goblin rolls the	for(int i=0; i<defenceNumberOfDiceRoles; randomDeffencePoints	Defense have value	Defense have value	PASS

dice x1 for defense	= rand()%defenceDiceSides+1; cout << randomDeffencePoints << " "; sum+=randomDeffencePoints; }			
Make sure Goblin gets values from 1-6 for each dice roll for defense	for(int i=0; i<defenceNumberOfDiceRoles; randomDeffencePoints = rand()%defenceDiceSides+1; cout << randomDeffencePoints << " "; sum+=randomDeffencePoints; }	Values of defense is between **1-6**	Values of defense is between **1-6**	PASS
Do similar tests described in Goblin for Barbarian. The difference is that barbarian needs to roll dice x2 for defense	for(int i=0; i<defenceNumberOfDiceRoles; randomDeffencePoints = rand()%defenceDiceSides+1; cout << randomDeffencePoints << " "; sum+=randomDeffencePoints; }	Values of defense is between **1-6**	Values of defense is between **1-6**	PASS
Make sure barbarian defence1 and defense values add up and to total defense	for(int i=0; i<defenceNumberOfDiceRoles; randomDeffencePoints = rand()%defenceDiceSides+1; cout << randomDeffencePoints << " "; sum+=randomDeffencePoints; }	defense will be equal to sum of defense1 and defense	defense will be equal to sum of defense1 and defense	PASS

points				
Do similar tests described for Goblin for Reptile (defense)	See golbin	See golbin	See golbin	PASS
Do similar tests described for Goblin for Blue Man	Similar to Goblin, the different is that, there have to be 3 rolls of dice	Make sure that are value for defense1, defense2 and defense3 and values are between **1-6**	Make sure that are value for defense1, defense2 and defense3 and values are between **1-6**	PASS
Make sure that appropriate class is instantiated when you use switch statement	cout statements to display the name, strength_ponts and armor for each character type. Strength will be displayed at the beginning of the battle. Armor will be displayed during the battle. Number of dice roles will be displayed during the battle	Compare values that were displayed to the value provided in the table- see above	Compare values that were displayed to the value provided in the table- see above	PASS

Make sure that resulted attack points are calculated correctly.	int netAttackPoints=player.attack()-enemy.deffence();
	cout << endl<< "Total damage - armor is " << netAttackPoints << " - " << enemy.getarmor();
	netAttackPoints= netAttackPoints - enemy.getarmor();
	cout << " = " << netAttackPoints <<end
		Values that were denerated by rolling dice (see above mentioned tests) are used in the formula, appropriate armor value is subtracted	Values that were denerated by rolling dice (see above mentioned tests) are used in the formula, appropriate	PASS

			armor value is subtracted	

Make sure that strength points do not go up during the battle when armor exceeded netAttack Points.	if(netAttackPoints<0){
	
	netAttackPoints=0;
	}
	There is a cout statement for netstrength
		Strength never go up	Strength never go up	PASS

Make sure that resulted strength points are calculated correctly	Cout statement with the net Strength points	At the end of the each battle – each time your press f, the resultant strength is calculated using strength- attack	At the end of the each battle – each time your press f, the resultant strength is calculated using strength-attack	PASS
Check that winner is announced when one of the characters runs out of strength_point	cout statement that print sstrength_points after each attack and defense round.	When health is 0 or less, winner/looser message is displayed	When health is 0 or less, winner/looser message is displayed	PASS
Check	cout statement that print	Whoever runs out	Whoever	PASS

that whoever runs out of strength point is labeled as looser	sstrength_points after each attack and defense round.	of strength_point first lost of the game	runs out of strength_point first lost of the game	
Check that if the player decides to eat a spinach, their attack points tripple	Cout statement of attack points	Attack points tripple	Attack points tripple	pass
Check that when enchanted weapon is used enchanted class is instantiated and attack point are double and player looses one strength point	cout statement for attack and cout statement of strength point	Attack x2 1 point from the strength is deducted	Attack x2 1 point from the strength is deducted	PASS
Check that then SaktiMans venom works correctly	Cout venom and attack points	4 points are added to the attack	4 points are added to the attack	PASS

Check that when players or enemy is a Mega Men a question is asked if they want to use poison	Cout question	Question is asked only for Mega Men and not for any other characters	Question is asked only for Mega Men and not for any other characters	PASS
If Mega Men has vials of poison and decides to use it, the deffence of the opponent is reduced by appropriate %- see design for % information	Cout defence	Poisoned deffence function will be called and deffence will be reduced by corresponding % for each character – defence value will be double	Poisoned deffence function will be called and deffence will be reduced by corresponding % for each character – defence value will be double	PASS
If the Mega men decides not to use poison a regular defence function	Count defense	Regular defense function is called and defense if not reduced – will look like an int	Regular defense function is called and defense if not reduced – will look like an int	PASS

is called				
If the mega man decides to use poison, but there are no poison vial left, a regular defense function is called	Cout that there are no poison left	Cout that there are no poison left Regular defense function is called Defense points will be an int	Cout that there are no poison left Regular defense function is called Defense points will be an int	PASS

TEST TO THE FOLLOWING BATTELS: PLEASE NOT THAT I WILL NOT TEST ALL THESE BATTLES BECAUSE I DO NOT SEE ANY EDUCATION VALUE IN THIS

CHRACTER	SPINACH	ENCHANTED WEAPON	POINSON	RESOLUTION OF THE BATTLE
PLAYER: GOBLIN	N	N	NA	Sometime character wins and sometimes enemy wins
ENEMY: GOBLIN	NA	NA	NA	Sometime character wins and sometimes enemy wins
PLAYER: GOBLIN	N	Y	NA	IF PLAYER USES ENCHANTED WEAPON ALL THE TIME, THE PLAYER WINS, IF ENCHANTED WEAPON USED RANDOMLY, THE OUTCOME VARIES, BUT MOST OF THE TIME ITS IN FAIVOR OF THE

				PLAYER
ENEMY: GOBLIN	NA	NA	NA	IF PLAYER USES ENCHANTED WEAPON ALL THE TIME, THE PLAYER WINS, IF ENCHANTED WEAPON USED RANDOMLY, THE OUTCOME VARIES, BUT MOST OF THE TIME ITS IN FAIVOR OF THE PLAYER
PLAYER: GOBLIN	Y	Y	NA	PLAYER WINS THE GAME
ENEMY: GOBLIN	NA	NA	NA	PLAYER WINS THE GAME
PLAYER: GOBLIN	Y	N	NA	PLAYER WINS THE GAME MOST OF THE TIME
ENEMY: GOBLIN	NA	NA	NA	PLAYER WINS THE GAME MOST OF THE TIME
PLAYER: GOBLIN	N	N	NA	PLAYER LOOSES THE GAME 99.9% OF THE TIME
ENEMY: BARBARIAN	NA	NA	NA	PLAYER LOOSES THE GAME 99.9% OF THE TIME
PLAYER: GOBLIN	N	Y	NA	IF PLAYER USES ENCHANTED WEAPON ALL THE TIME, THE PLAYER WINS, IF ENCHANTED WEAPON USED RANDOMLY, THE OUTCOME VARIES, BUT MOST OF THE

				TIME ITS IN FAIVOR OF THE PLAYER
ENEMY: BARBARIAN	NA	NA	NA	IF PLAYER USES ENCHANTED WEAPON ALL THE TIME, THE PLAYER WINS, IF ENCHANTED WEAPON USED RANDOMLY, THE OUTCOME VARIES, BUT MOST OF THE TIME ITS IN FAIVOR OF THE PLAYER
PLAYER: GOBLIN	Y	Y	NA	PLAYER WINS 99.9% OF THE TIME
ENEMY: BARBARIAN	NA	NA	NA	PLAYER WINS 99.9% OF THE TIME
PLAYER: GOBLIN	Y	N	NA	ENEMY LOOSES A LOT
ENEMY: BARBARIAN	NA	NA	NA	ENEMY LOOSES A LOT
PLAYER: BARBARIAN	N	N	NA	ENEMY LOOSES 99% OF THE TIME
ENEMY: GOBLIN	NA	NA	NA	ENEMY LOOSES 99% OF THE TIME
PLAYER: BARBARIAN	N	Y	NA	IF PLAYER USES ENCHANTED WEAPON ALL THE TIME, THE PLAYER WINS, IF ENCHANTED WEAPON USED RANDOMLY, THE OUTCOME

				VARIES, BUT MOST OF THE TIME ITS IN FAIVOR OF THE PLAYER
ENEMY: GOBLIN	NA	NA	NA	IF PLAYER USES ENCHANTED WEAPON ALL THE TIME, THE PLAYER WINS, IF ENCHANTED WEAPON USED RANDOMLY, THE OUTCOME VARIES, BUT MOST OF THE TIME ITS IN FAIVOR OF THE PLAYER
PLAYER: BARBARIAN	Y	Y	NA	PALYER WINS 99.9% OF THE TIME
ENEMY: GOBLIN	NA	NA	NA	PLAYER WINS 99.9% OF THE TIME
PLAYER: BARBARIAN	Y	N	NA	PALYER WINS 99.9% OF THE TIME
ENEMY: GOBLIN	NA	NA	NA	PLAYER WINS 99.9% OF THE TIME
PLAYER: GOBLIN	N	N	NA	PLAYER LOOSES THE GAME 99.9% OF THE TIME
ENEMY: REPTILE	NA	NA	NA	PLAYER LOOSES THE GAME 99.9% OF THE TIME
PLAYER: GOBLIN	N	Y	NA	IF PLAYER USES ENCHANTED WEAPON ALL THE TIME, THE PLAYER WINS, IF

				ENCHANTED WEAPON USED RANDOMLY, THE OUTCOME VARIES, BUT MOST OF THE TIME ITS IN FAIVOR OF THE PLAYER
ENEMY: REPTILE	NA	NA	NA	IF PLAYER USES ENCHANTED WEAPON ALL THE TIME, THE PLAYER WINS, IF ENCHANTED WEAPON USED RANDOMLY, THE OUTCOME VARIES, BUT MOST OF THE TIME ITS IN FAIVOR OF THE PLAYER
PLAYER: GOBLIN	Y	Y	NA	PLAYER WINS 99.9% OF THE TIME
ENEMY: REPTILE	NA	NA	NA	PLAYER WINS 99.9% OF THE TIME
PLAYER: GOBLIN	Y	N	NA	BOTH ENEMY AND PLAYER CAN WIN
ENEMY: REPTILE	NA	NA	NA	BOTH ENEMY AND PLAYER CAN WIN
PLAYER: REPTILE	N	N	NA	ENEMY LOOSES 99% OF THE TIME
ENEMY: GOBLIN	NA	NA	NA	ENEMY LOOSES 99% OF THE TIME
PLAYER:	N	Y	NA	ENEMY LOOSES

REPTILE				99% OF THE TIME
ENEMY: GOBLIN	NA	NA	NA	ENEMY LOOSES 99% OF THE TIME
PLAYER: REPTILE	Y	Y	NA	ENEMY LOOSES 99% OF THE TIME
ENEMY: GOBLIN	NA	NA	NA	ENEMY LOOSES 99% OF THE TIME
PLAYER: REPTILE	Y	N	NA	ENEMY LOOSES 99% OF THE TIME
ENEMY: GOBLIN	NA	NA	NA	ENEMY LOOSES 99% OF THE TIME
PLAYER: GOBLIN	N	N	NA	
ENEMY: BLUE MAN	NA	NA	NA	
PLAYER: GOBLIN	N	Y	NA	
ENEMY: BLUE MAN	NA	NA	NA	
PLAYER: GOBLIN	Y	Y	NA	
ENEMY: BLUE MAN	NA	NA	NA	
PLAYER: GOBLIN	Y	N	NA	
ENEMY: BLUE MAN	NA	NA	NA	
PLAYER: BLUE MAN	N	N	NA	
ENEMY: GOBLIN	NA	NA	NA	
PLAYER: BLUE MAN	N	Y	NA	
ENEMY: GOBLIN	NA	NA	NA	
PLAYER:	Y	Y	NA	

BLUE MAN				
ENEMY: GOBLIN	NA	NA	NA	
PLAYER: BLUE MAN	Y	N	NA	
ENEMY: GOBLIN	NA	NA	NA	
PLAYER: GOBLIN	N	N	NA	
ENEMY: MEGA MAN	NA	NA	N	
PLAYER: GOBLIN	N	N	NA	
ENEMY: MEGA MAN	NA	NA	Y	
PLAYER: GOBLIN	N	Y	NA	
ENEMY: MEGA MAN	NA	NA	N	
PLAYER: GOBLIN	N	Y	NA	
ENEMY: MEGA MAN	NA	NA	Y	
PLAYER: GOBLIN	Y	Y	NA	
ENEMY: MEGA MAN	NA	NA	N	
PLAYER: GOBLIN	Y	Y	NA	
ENEMY: MEGA MAN	NA	NA	Y	
PLAYER: GOBLIN	Y	N	NA	
ENEMY: MEGA MAN	NA	NA	N	
PLAYER: GOBLIN	Y	N	NA	
ENEMY: MEGA MAN	NA	NA	Y	
PLAYER: MEGA MAN	N	N	N	
ENEMY:	NA	NA	NA	

GOBLIN				
PLAYER: MEGA MAN	N	N	Y	
ENEMY: GOBLIN	NA	NA	NA	
PLAYER: MEGA MAN	N	Y	N	
ENEMY: GOBLIN	NA	NA	NA	
PLAYER: MEGA MAN	N	Y	Y	
ENEMY: GOBLIN	NA	NA	NA	
PLAYER: MEGA MAN	Y	Y	N	
ENEMY: GOBLIN	NA	NA	NA	
PLAYER: MEGA MAN	Y	Y	Y	
ENEMY: GOBLIN	NA	NA	NA	
PLAYER: MEGA MAN	Y	N	N	
ENEMY: GOBLIN	NA	NA	NA	
PLAYER: MEGA MAN	Y	N	Y	
ENEMY: GOBLIN	NA	NA	NA	
PLAYER: GOBLIN	N	N	NA	
ENEMY: SATKI MAN	NA	NA	NA	
PLAYER: GOBLIN	N	Y	NA	
ENEMY: SATKI MAN	NA	NA	NA	
PLAYER: GOBLIN	Y	Y	NA	
ENEMY: SATKI MAN	NA	NA	NA	
PLAYER:	Y	N	NA	

GOBLIN				
ENEMY: SATKI MAN	NA	NA	NA	
PLAYER: SATKI MAN	N	N	NA	
ENEMY: GOBLIN	NA	NA	NA	
PLAYER: SATKI MAN	N	Y	NA	
ENEMY: GOBLIN	NA	NA	NA	
PLAYER: SATKI MAN	Y	Y	NA	
ENEMY: GOBLIN	NA	NA	NA	
PLAYER: SATKI MAN	Y	N	NA	
ENEMY: GOBLIN	NA	NA	NA	
PLAYER: BARBARIN	N	N	NA	
ENEMY: BARBARIAN	NA	NA	NA	
PLAYER: BARBARIN	N	Y	NA	
ENEMY: BARBARIAN	NA	NA	NA	
PLAYER: BARBARIN	Y	Y	NA	
ENEMY: BARBARIAN	NA	NA	NA	
PLAYER: BARBARIN	Y	N	NA	
ENEMY: BARBARIAN	NA	NA	NA	
PLAYER: REPTILE	N	N	NA	
ENEMY: REPTILE	NA	NA	NA	
PLAYER: REPTILE	N	Y	NA	
ENEMY:	NA	NA	NA	

REPTILE				
PLAYER: REPTILE	Y	Y	NA	
ENEMY: REPTILE	NA	NA	NA	
PLAYER: REPTILE	Y	N	NA	
ENEMY: REPTILE	NA	NA	NA	

ETC……

At this point, I decided that I am wasting time. I have not listed all characters combinations, but I do not see any value in what I am doing here. Thus, I'll stop it and do something more productive. There is no educational value in what I am doing here

Reflection:

1. So initially, I was planning to make attack and defense function pure virtual, make class character and basically copy/past attack and defense functions into each monster class. However, I decided that it would be too much code duplication because functions attack and defense are basically the same for all character types.

The requirements of the assignment is that the parent class needs to be an abstract class. So what I did instead of coping and pasting attack/defense functions into each monster class, I decided to create a different parent class, creature and make my character class (the one that was the parent class on the previous assignment) to be a child class of this new creature class.

```
class CREATURE{
protected:
        string name;//name of the charector
        int atackNumberOfDiceRoles;
        int atackDiceSides;//Dice sides for attack
        int armor;
        int defenceNumberOfDiceRoles;
        int defenceDiceSides;
        int damage;
public:
        int getarmor();
        virtual int attack()=0;
        int deffence();
        int getdamage();
        string getname();
};
```

And my character class became:

```
class CHARECTOR:public CREATURE{
protected:
        int spinach; THIS IS A NEW VARIABLE BECAUSE I AM INTRODUCING A SPINACH
        int strength;//dynamically change strength power
        int ststore;// this variable will be used to calculate the % of the strenght points left
public:
int getarmor(){return(armor);}
int getstrength(){return(strength);}
void setstrength(int strength2){  strength=strength2;  }
 virtual int attack() THE ATTACK FUNCTION WILL BE THE SAME AS IT WAS ON THE PREVIOUS ASSIGNMENT.
 int deffence(){
    int getstnstore(){return(ststore);}
    int getdamage(){return(damage);}
   string getname(){return(name);}
   int getSpinach(){return(spinach);} THIS IS A NEW FUNCTION, WE THE DEFINITION BELOW
};
```

2. Once the character selected, the player will be asked if they want to eat spinach at the beginning of each attack. Please note there will be another change here to the original design. Initially, I was planning to let both the players and the enemies eat the spinach and use enchanted weapon. However, when I tried to calculate how much time I would need to do the testing and test each battle combination, I realized that I do not have enough time to test all combinations. As the result of this, only the player will be able to eat the spinach and use enchanted weapon. At the beginning of each attack, the player will be asked if they want to each the spinach, is the player says yes, the attack will be stronger, by the factor of 3 (I decided that quadrupling the attack was way too much). Another change in the design is that

I placed spinach in the parent class and not into each monster class. The constructor in each monster classes got variable spinach initialized to 3. Thus, if the player decides to eat spinach, attack points will triple.

3. I decided not to add any functions to the old monsters: GOBLIN, BARBARIAN, REPTILE AND BLUE MAN. This was mainly done to avoid spending hours testing. Thus, the only difference in these characters would be the spinach.
4. Next, I created 2 new classes of monsters:
    a. class MegaMan:public Barbarian is the child of barbarian
    b. class SaktiMan:public Goblin is the child of the goblin

This is another change to the original design. Initially, I was so ambisions that I was planning to add venom to all my charactes. However, as I already pointed out multiple times, I have not estimated before how long it would take me to test my game. So there will be a lot of revisions to my original design. Back to the MegaMan and SaktiMan. Only these 2 characters will be able to use venom. Also, I decided to do additional reading on venoms and found out that there are various poisons (not only ones that just do damage over time, but also weakening venoms that decrease attack, movement speed, damage output or resistance).
http://forum.worldofplayers.de/forum/archive/index.php/t-306829-p-2.html
Mega Man will use poison, as described in design. Mega Men will have 2 vials of poison and there will be a questing that would ask if he wants to use a poison. If the answer is yes and poison is available, the opponent in the game will have weaker defense. I decided that the venom will not do anything with the attack point. As a result of this, I've added new functions to each monster class:

```
//CLASS GOBLIN
class Goblin:public CHARECTOR{
   virtual double defencePoisoned(){
         double poisonedDefense;
         poisonedDefense = deffence();
         poisonedDefense = 0.5*poisonedDefense;

      cout<<"Sum : " << poisonedDefense << endl;// message that will indicate total attack points

                  return(poisonedDefense);
      }
};

///CLASS BARBARIAN
class Barbarian:public CHARECTOR{
         poisonedDefense = 0.7 * deffence();
///CLASS REPTILEPEOPLE
class ReptilePeople:public CHARECTOR{
         poisonedDefense = 0.8* deffence();
///CLASS BLUEMEN
class BlueMen:public CHARECTOR{
         poisonedDefense = 0.9 * deffence();
```

Because of this change, Ill be working with doubles and thus, I need to change all my int variables for defence and attack to double type.

5. SaktiMan man on the other hand will use venom in a different way. Any time, the SaktiMan attacks, venom would add to the attack points. Thus, I need to overwrite attack function for SaktiMan:

```
int attack(){
 int sum=0;
 int randomAttackPoints;// variable to keep track of atack points
 cout << endl << name << " - attack ";// will display a message who is
attacking
 for(int i=0; i<atackNumberOfDiceRoles; i++){// the dice will be rolled as
many times as indicated for each character
    randomAttackPoints = rand()%atackDiceSides + 1; // the values of atack
will be between 1 and sides of atack points
    cout << randomAttackPoints << " ";// message that will display how many
atack points are generated
    sum+=randomAttackPoints;
 }
 cout<<"Sum : " << sum <<"+ Venom : "<< venom<<"Total"<<(sum+venom)
<<endl;// message that will indicate total attack points
     sum=sum+venom;
 return(sum);
}
```

At the same time constructor will have variable venom initialized to 4, which is # of points that will be added to the atackpoinst to calculate the total damage of attack.

6. Because of the venom, I have to add few simple functions:
   a. For Mega Man

```
int getPoison(){return(poisonVials);}
void setPoison(){
     int poisonVials1;
     poisonVials--;
     poisonVials1= poisonVials;}
```

   b. For SaktiMan:

```
int getVenom(){return(venom);}
```

7. There was a revision to the enchanted weapons as well. Initially, I was planning to let the characters use enchanted weapons if they had energy points. To simplify my program, they would be able to use enchanted weapons any time, if they answer yes to the question: Do you want to use enchanted weapon. If they decide to use enchanted weapon, the following class will be called:

```
//CLASS ENCHANTEDHOBBIT for double damage
class EnchantedHobbit{
private:
  int doubleDamage;
public:
  EnchantedHobbit(){doubleDamage=2;}
  int getEnchantedHobbit(){return(doubleDamage);}
};
```

Any time the enchanted weapon is used, there is a double damage to the enemy. Only player will be able to use enchanted weapon. There will be a drawback using enchanted weapon. If the player decides to use enchanted weapon, 1 point will be subtracted from the strength of the player
netAttackPoints=netAttackPoints*e.getEnchantedHobbit();
playerNewStrength=player->getstrength()-1; // if enchanted weapon is used 1 point from the strength is deducted
if(playerNewStrength<0){playerNewStrength=0;}//set 0 for -ve strength
player->setstrength(playerNewStrength);}

8. Finally, last change that I decided to do is to redefine deffence function of the Mega Man.

```
double deffence(){
    int sum=0;
    int randomDeffencePoints;// variable to keep track of deffence points
    cout << endl << name << " - defence ";// will display a message who is defending

    for(int i=0; i<defenceNumberOfDiceRoles; i++){// the dice will be rolled as many times as indicated for each character
    randomDeffencePoints = rand()%defenceDiceSides+1;// the values of defence will be between 1 and sides of defence points
        cout << randomDeffencePoints << " "; // message that will display how many defence points are generated
        sum+=randomDeffencePoints;
    }

  if ((rand()%100 +1) >50){  50% of the time sum will be returned
     sum = sum;
  }
  else{
      sum = sum/2; //otherwise, sum/2 will returned


    cout<<"Sum : "<<sum<<endl;// message that will indicate total defence points

        return(sum);
  }
```

I spend so much time, just to writing down what needs to be tested and then I started testing deferent combat, realized that everything works as expected and did not see the value of additional testing.