Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
Capitalized	1	Lowerword contained more or less than 5 letters	hot, amazing	HOT, AMAZING	HOT, AMAZING	P
	2	Lowerword is a 5 Word containing both Uppercase and lowercase	gRanD, AarGh	GRAND, AARGH.	GRAND, AARGH	P
	3	Lowerword is only 5 small letters	grand, aargh	GRAND, AARGH	GRAND, AARGH	P

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
CheckLetter	1	The word has special characters	!@%^*	0	0	P
	2	Word has upper and lower case	gRanD, AarGh	1	1	P
	3	word is only 5 small letters	grand, aargh	1	1	P

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
CheckLength	1	Word is bigger than 5	Awesome	0	0	P
	2	Word is special character but it is len of 5	w@t3r	1	1	P
	3	word is only 5 small letters	grand, aargh	1	1	P

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
ValidWord	1	Input contained	hot,	0,0	0,0	P

		more or less than 5 letters	amazing			
	2	Input is a 5 Word containing both Uppercase and lowercase	gRanD, AarGh	1,1	1,1	P
	3	Input is only small letters	grand, aargh	1,1	1,1	P

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
MainGame	1	Guess contained more or less than 5 letters	Banana, Come	The word is not valid Enter a new one	The word is not valid Enter a new one	P
	2	Player is right on the 5th attempt	Grand, Inept, Slurp, Aargh, (right word)	You got it right the word is (right word) & return 5	You got it right the word is (right word) & return 5	P
	3	Player didn't get it right on 6th attempt	Grand, Inept, Slurp, Aargh, Aegis, Begun	You did not get it right , the right word is (right word) & return -1	You did not get it right , the right word is (right word) & return -1	P

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
EnterManual	1	PickW contained more or less than 5 letters	hot, amazing	Error, Word is not part of the dictionary can't be used	Error, Word is not part of the dictionary can't be used	P
	2	PickW is a 5 letter but, is not real word	AAAAA, abcde	Error, Word is not part of the dictionary can't be used	Error, Word is not part of the dictionary can't be used	P

	3	PickW is an allowed word	Great	GREAT	GREAT	P

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
CheckProfile	1	ABC is a real profile and is in Masterlist[0], return position of ABC	Name contain: ABC	0	0	P
	2	Name is not a profile	Name contain: ZZZ, BBB	-1	-1	P
	3	Invalid word length	Name contain: John	-1	-1	P

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
CheckSpace	1	There is “\0” in MasterList, if Score.txt doesn’t exist	N/A	0	0	P
	2	Score.txt exists but there is only some profile (only 2 profile exists)	N/A	2	2	P
	3	Score.txt exists but it’s full (20 profiles exists)	N/A	-1	-1	P

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
CheckName	1	Name has invalid amount of letters	Lame	Invalid Name	Invalid Name	P
	2	Name is all small 3 letters	ccc	CCC	CCC	P

	3	Name is a mix of letters and special character	!!a	Invalid Name	Invalid Name	P

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
WinIncrease	1	MasterList.Streak is higher than MasterList.BStreak	MasterList.BStreak = 5 MasterList.Streak = 10	MasterList.BStreak = 10	MasterList.BStreak = 10	P
	2	MasterList.time is higher than time	MasterList.time = 60 Time = 30	MasterList.time = 11	MasterList.time = 11	P
	3	MasterList.Streak is lower than MasterList.BStreak	MasterList.BStreak = 10 MasterList.Streak = 5	MasterList.BStreak = 10	MasterList.BStreak = 10	P

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
TotalFreq	1	If 2 or more Masterlist.Frequency have values	MasterList[0].Frequency = {10,10,10, 10,10,10} MasterList[1].Streak = {10, 10, 10, 10, 10,10}	Total 1-6 win/s attempts: 20	Total 1 -6 win/s attempts: 20	P
	2	If MasterList does not have values or no players	MasterList[0-19].Frequency[0-6] = 0	Total 1-6 win/s attempts: 0	Total 1-6 win/s attempts: 0	P
	3	If a player exist	MasterList[0].Frequency = {5,10,20,30,40,50}	Total 1-6 win/s attempts: {5, 10, 20, 30, 40 ,50}	Total 1-6 win/s attempts: {5, 10, 20, 30, 40 ,50}	P

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
PrintBStreak	1	Array of MasterList[].BStreak is in a increasing order	MasterList[0-9].BStreak = {3, 5, 7, 9, 20, 35, 47, 53, 60, 70}	MasterList[9-5].BStreak = {70, 60 , 53, 47 ,35}	MasterList[9-7].BStreak = {70, 60 , 53}	P
	2	Array of MasterList[].BStreak is in a decreasing order	MasterList[0-9].BStreak = {70, 60, 53, 47, 35, 20, 9, 7, 5, 3}	MasterList[0-4].BStreak = {70, 60 , 53}	MasterList[0-4].BStreak = {70, 60 , 53}	P
	3	Array of MasterList[].BStreak is in a random order	MasterList[0-9].BStreak = {53, 5, 7, 60, 9, 35, 70, 3, 20, 47}	Prints = {70, 60 , 53}	Prints = {70, 60, 53}	P

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
BStreakName	1	Array of MasterList[].BStreak is in a increasing order	MasterList[0-9].BStreak = {3, 5, 7, 9, 20, 35, 47, 53, 60, 70}	MasterList[9-5].BStreak = {70, 60 , 53, 47 ,35}	MasterList[9-5].BStreak = {70, 60 , 53, 47 ,35}	P
	2	Array of MasterList[].BStreak is in a decreasing order	MasterList[0-9].BStreak = {70, 60, 53, 47, 35, 20, 9, 7, 5, 3}	MasterList[0-4].BStreak = {70, 60 , 53, 47 ,35}	MasterList[0-4].BStreak = {70, 60 , 53, 47 ,35}	P
	3	Array of MasterList[].BStreak is in a random order	MasterList[0-9].BStreak = {53, 5, 7, 60, 9, 35, 70, 3, 20, 47}	Prints = {70, 60 , 53, 47 ,35}	Prints = {70, 60, 53, 47, 35}	P

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
Besttime	1	Array of MasterList[].time is in a increasing order	MasterList[0-9].time = {10, 20, 30, 40, 50, 60 , 70, 80 ,90, 100}	MasterList[0-4].time = {10, 20 , 30, 40 ,50}	MasterList[9-5].time ={10, 20 , 30, 40 ,50}	P
	2	Array of MasterList[].time is in a decreasing order	MasterList[0-9].time = {100, 90, 80, 70, 60, 50, 40, 30, 20, 10}	MasterList[9-5].time = {10, 20 , 30, 40 ,50}	MasterList[9-5].time = {10, 20 , 30, 40 ,50}	P
	3	Array of MasterList[].time is in a random order	MasterList[0-9].time = {30, 40, 70, 10, 100, 20, 80, 90, 50, 10}	Prints {10, 20 , 30, 40 ,50	Prints = {10, 20, 30, 40, 50}	P

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
BestAttem pts	1	Array of MasterList[].Frequ ency[nAttempts] in an increasing order	MasterList[0-9].Frequ ency[nAtte mpts] = {10, 20, 30, 40, 50, 60 , 70, 80 ,90, 100}	MasterList[0-4].Frequ ency[nAtte mpts] = {100, 90, 80, 70 ,60}	MasterList[9-5].Frequ ency[nAtte mpts] ={100, 90, 80, 70 ,60}	P
	2	Array of MasterList[].Frequ ency[nAttempts] is in a decreasing order	MasterList[0-9].time = {100, 90, 80, 70, 60, 50, 40, 30, 20, 10}	MasterList[0-4].Frequ ency[nAtte mpts] = {100, 90, 80, 70 ,60}	MasterList[0-4].Frequ ency[nAtte mpts] = {100, 90, 80, 70 ,60}	P
	3	Array of MasterList[].Frequ ency[nAttempts] is in a random order	MasterList[0-9].Frequ ency[nAtte mpts] = {20, 30, 10, 70, 50, 60,	Prints {100, 90 , 80, 70 ,60	Prints = {100, 90, 80, 70, 60}	P

			40, 90, 100, 80}			

Function	#	Description	Sample INPUT	Expected Output	Actual Output	P/F
ViewStats	1	If a player input within a bound no. 1, and enter a valid name. ABC stats: Streak = 5 BStreak = 15 Best Time = 30 Frequency[0] = 2 Frequency[1] = 8 Frequency[2] = 14 Frequency[3] = 20 Frequency[4] =22 Frequency[5] = 30	nSubDec = 1 Name:ABC	Name: ABC Streak: 5 Best Streak: 15 Best Time: 30 Frequency: 1 win/s Attempts: 2 2 win/s Attempts: 8 3 win/s Attempts: 14 4 win/s Attempts: 20 5 win/s Attempts: 22 6 win/s Attempts: 30	Name: ABC Streak: 5 Best Streak: 15 Best Time: 30 Frequency: 1 win/s Attempts: 2 2 win/s Attempts: 8 3 win/s Attempts: 14 4 win/s Attempts: 20 5 win/s Attempts: 22 6 win/s Attempts: 30	P
	2	Player Enter Invalid number	nSecDec = 4	Invalid Input	Invalid INput	P
	3	Player enter 3, and player enter a submenu but , inputted ainvalid value	nRankDec =4	Invalid Input	Invalid Input	P