Genetic Algorithm: class documentation – Chromosome class

CLASS NAME	**Cchromosome**	
Data Members	Access	Description
unsigned char* genes	private	Each array element holds 8 alleles (8 bit data type)
unsigned int arraySize	private	Holds the size of the array needed to store the whole gene-string (the number of 8-bit elements in the *genes* array)
unsigned int fitness	private	Stores the current *fitness score* assigned to the chromosome
unsigned int length	private	Length of the gene string (in *bits*) –all chromosomes in a given *population* should have the same number of genes

Return Value	Function Name	Parameters	Description	Notes
	default constructor			
	custom constructor	unsigned int newlength	The total # of alleles (bits)	
	copy constructor	const chromosome& copyChromosome		
	destructor			
void	**AssignFitness**	unsigned int newfitness	The new *Fitness Score* to be assign to the chromosome	The *Fitness Score* is calculated by a *Fitness Function* (defined in the main program).
BOOL *(int)* 0 = ERROR 1 = SUCCESSFUL COPY	**Copy**	chromosome* received	pointer to another chromosome object	*this* chromosome's and the passed chromosome's gene string must be of equal length for the copy to work.
void	**CreateGenes**	unsigned int newlength	The length of the gene string (in *bits*)	Sets up and DMAs the gene array. Assigns a random value to each *bit* (0 or 1). If default constructor is called then this needs to be called explicitly.
void	**DisplayFitness**			DOS-only
void	**DisplayGenes**			DOS-only
int the decimal value of the extracted portion of the bit string (always +). -1 signifies that no value could be returned-ERROR.	**ExtractValue**	unsigned int position	bit position of the *high order bit* that forms the extracted value.	Calculates and returns the decimal value from a defined portion of the gene(bit) string. The returned value is always a positive integer. The maximum value that can be returned is: 32767.
		unsigned int valsize	The # of bits in the extracted portion of the bit string. Max valsize is 15 bits.	
unsigned int Returns the value of *fitness* (private data member).	**GetFitness**			the *Fitness Score* of the chromosome.
unsigned int Returns the value of *length* (private data member).	**GetLength**			The bit length of the gene string.
BOOL *(int)* 0 = ERROR 1 = SUCCESSFUL INSERTION	**InsertValue**	unsigned int position	the position of the the *high order bit* of the inserted value.	This is the reverse of *ExtractValue.* The user assigns the position in the bit string where the decimal value is to be inserted and the number of bits allocated to storing this value. The function then attempts to insert the binary equivalent of the decimal value into the bit string (overwriting any bit values occupying the allocated space).
		unsigned int bitlength	The # of *bits* allocated to representing the inserted value (increased by the function if more bits are required to hold the inserted value).	
		unsigned int value	The value (passed in decimal) to be represented in the allocated portion of the bit string.	

Return Value	Function Name	Parameters	Description	Notes
int 0 = CAN'T MATE 1 = CROSSOVER 2 = NO CROSSOVER	**Mate**	chromosome* partner	A pointer to the chromosome, which is to be mated with *this.*	Attempts to mate *this* chromosome with a selected *partner* chromosome, which become the 2 *children* chromosomes to be used in the next generation. Crossover probability is passed to the function. Chromosomes must have the same number of genes in order to mate.
		int crossProb	The probability of genetic crossover (the default value of 800 equates to 80%)	
void	**Mutate**	int mutaProb	The probability of mutation of each *bit* in the gene string. (the default value of 5 equates to 0.5%)	The function moves through each *bit* of the gene string and decides whether it should be flipped (or mutated) according to the passed probability of mutation.
void	**Randomise**			Re-randomises the entire gene string of a chromosome (the gene string is initially randomised when it is created).

Genetic Algorithm: class documentation – Population class

CLASS NAME	**Cpopulation**			
Data Members	Access	Description		
const unsigned int size	private	Total number of chromosomes in the population		
const unsigned int chromolen	private	Length to be assigned to all chromosomes in a population (default value = 32)		
unsigned int generation	private	*Generation* counter (incremented by *NextGeneration*)		
unsigned int mutaProb	private	*Mutational Probability* to be used in population's mutation phase (default value = 5 (0.5% probability of any gene mutating))		
unsigned int crossProb	private	*Cross-over Probability* of chromosomes in the population (default value = 800 (80% probability of gene cross-over))		
chromosome* currentPool	private	Pointer to first chromosome in current population		
chromosome* matingPool	private	Pointer to first chromosome in the *selected chromosomes* pool		
unsigned int fitness	private	Overall fitness of population (relates to the chromosomes in the *currentPool* only)		
float meanFitness	private	The average fitness of a chromosome in the population		
unsigned int elitism	private	# of Elite chromosomes to be selected from the *current pool* (0 = NO ELITISM)		
unsigned int swapRate	private	(Multi-Point Crossover ONLY) probability of gene swap at the current locus. (default value = 500 (50% probability of gene swap over))		
unsigned int eliteCopies	private	# of copies to be made of each *Elite Chromosome*		
Return Value	Function Name	Parameters	Description	Notes
	default constructor			
	custom constructor	unsigned int newsize	The total # of *chromosomes* in the population pool (must be an even number)	All chromosomes in a population are created with the same number of genes (to allow successful crossover). The population size must be even so that all chromosomes in the 'mating pool' can be paired up.
		unsigned int newchromolen	The length of a *chromosome's* gene-string (in bits)	
		unsigned int newcrossProb	The *Cross-over probability*	
		unsigned int newmutaProb	The *Mutational Probability*	
		unsigned int newelitism	# of Elite Chromosomes	
		unsigned int neweliteCopies	# of copies of Elite Chromosomes	
		usigned int newswapRate	Multi-Point gene crossover rate	
	copy constructor			
	destructor			
void	**AssignFitness**	unsigned int element	index of the chromosome	The *Fitness Score* is calculated by a *Fitness Function* (defined in the main program).
		unsigned int newfitness	The new fitness value of the chromosome	
void	**DisplayCurrent**			Display data on the current population (DOS-only).
void	**DisplayMating**			Display data on the selection pool (DOS-only).

Return Value	Function Name	Parameters		Description	Notes
int the decimal value of the extracted portion of the bit string (always +). -1 signifies that no value could be returned-ERROR.	ExtractValue	unsigned int element		The chromosome index in the in the *currentPool* array	Calculates and returns the decimal value from a defined portion of the gene(bit) string, of a particular chromosome in the *currentPool*. The returned value is always a positive integer. The maximum value that can be returned is: 32767.
		unsigned int position		bit position (within the chromosome) of the *high order bit* that forms the extracted value.	
		unsigned int valsize		The # of bits in the extracted portion of the bit string. Max valsize is 15 bits.	
unsigned int Returns the value of *chromolen* (private data member).	GetChromoLen				Returns the length of chromosomes in the population.
unsigned int Returns the value of *crossProb* (private data member).	GetcrossProb				Returns the cross-over probability for the population
unsigned int Returns the value of *generation* (private data member).	GetGeneration				Returns the current generation count
unsigned int Returns the value of *mutaProb* (private data member).	GetmutaProb				Returns the mutational probability value for chromosomes in the population.
unsigned int Returns the value of *size* (private data member).	GetSize				Returns the number of chromosomes in the population (this is always an even number and is constant throughout).
BOOL *(int)* 0 = ERROR 1 = SUCCESSFUL INSERTION	InsertValue	unsigned int element		The chromosome index in the in the *currentPool* array	This is the reverse of *ExtractValue.* Inserts a value into the gene-string of a chromosome.
		unsigned int position		the position of the the *high order bit* of the inserted value (within the chromosome).	
		unsigned int bitlength		The # of *bits* allocated to representing the inserted value (increased by the function if more bits are required to hold the inserted value).	
		unsigned int value		The value (passed in decimal) to be represented in the allocated portion of the bit string.	

Return Value	Function Name	Parameters	Description	Notes
void	**Mate**	unsigned int method	0 = Multi-Point Crossover 1 = Single-Point 2 = Two-Point (default = 1)	Mate all chromosomes in selection pool to produce the next generation's (children) chromosomes.
void	**Mutate**			Run the mutation process on each chromosome
void	**NextGeneration**			Creates the next generation of chromosome objects: copies the *matingPool* (after mating/mutation has taken place) back into the *currentPool*, and increment the *generation* counter.
void	**Select**	unsigned int method	0 = Roulette Wheel Selection 1 = Tournament Selection	In both selection methods, the fittest chromosomes have the highest probability of being selected for the *matingPool.* The population's total *fitness* and the *fitness* of each chromosome (in the current population) needs to have been determined by an external *Fitness Function* before this is called.
		unsigned int sample	# of chromosomes randomly 'sampled' (used in Tournament Selection).	