Due Date: 01.12.2023

CENG311 HW2

In this homework assignment, you will implement array of playlists with MIPS assembly language. Each playlist is a simple linked list, with each node's data pointing to a song structure. You are expected to modify the source code provided with this homework.

- You are expected to fill given subroutines with the usages at the Subroutine Usages table.
- In the main subroutine, you should call the other subroutines according to expected operations:
  - Initialize array of playlist by using with an initial size of 3.
  - Create 3 playlists as linked list and insert them into array of playlist.
  - Resize to size of 5. Create 2 more playlists and insert them into available areas in array of playlist.
  - Create 4 songs for each playlist and insert them into playlists. Each song must have a name and duration.
  - Print all the songs.
  - Remove the second song of each playlist.
  - Print all the songs.
  - Add 1 more song to each playlist.
  - Print all the songs.
  - Remove the fourth playlist.
  - Print all the songs.
  - Search for given two songs, print the result.
- Hint: you must print and search songs with traverse subroutines. Traverse subroutines should be used to go through to operate on each array or linked list element.
- Generic subroutines should be implemented for general operations. For example, linked list related subroutines should only include linked list related operations. You should not include any song logic to generic subroutines.
- All variables needed has been declared. Therefore, you should not define any new variable.
- Structures of the array, linked list, and song is given.
- Song name variables are given with the format "PlaylistNumberSongNumber" Ex: p1s1. Durations are also given with the format "PlaylistNumberSongNumber_duration" Ex: p1s1_duration.
- Two songs that will be searched is given as search1 and search2.
- You can allocate heap memory areas by using syscall with the sbrk code 9 ($v0 -> 9, $a0 -> memory size).
- There is no freeing for MIPS, so you should not try to free the spaces you allocated.
- You should not use the variable sReg in your instructions.
- S registers are expected to remain unchanged after the subroutines are used.
- Each subroutine should be implemented.
- Homework will be evaluated using Mars MIPS simulator.
- Change the assembler source code name to your student id, and upload it e.g., 280000000.asm
- You must only upload the .asm assembler source code file.
- No collaboration is allowed.

Structures:

Array	
4 Bytes - Address of the Data	4 Bytes - Size
Linked List Node	
4 Bytes - Address of the Data	4 Bytes - Address of the Next Node
Song	
4 Bytes - Address of the Name (name itself is 64 bytes)	4 Bytes - Duration

Subroutine Usages:

Subroutine Name	Argument 1 ($a0)	Argument 2 ($a1)	Argument 3 ($a2)	Return Value ($v0)
GENERIC				
createArray	Size of the array	-	-	Address of the array
resizeArray	Address of the array	Old Size	New Size	Address of the array
putElementToArray	Address of the array	Index	Address of the element	-
removeElementFromArray	Address of the array	Index	-	-
createLinkedList	-	-	-	Address of the linked list
putElementToLinkedList	Address of the linked list	Address of the element	-	-
removeElementFromTheLinkedList	Address of the linked list	Index	-	-
traverseArray	Address of the function (isSong, printSong)	Address of the array (it includes starting address and size)	-	-
traverseLinkedList	Address of the function (isSong, printSong)	Address of the linked list	-	-
NON-GENERIC				
createSong	Address of the song name	Duration of the song	-	Address of the song
isSong	Address of the song	-	-	-
compareString	Address of the first string	Address of the second string	Comparison size	1 for found, 0 for not found
printSong	Address of the song	-	-	-