Ονοματεπώνυμο: Παντελεήμων Μαλέκας
A.M: 1115201600268

For the first project, I implemented the functions in heap_file.c. Here are my design choices about each function:

1.) HP_Init()

I didn't add anything in this function since it wasn't necessary.

2.) HP_CreateFile(const char* filename)

In this function I create a heap file using the functions from the BF_Block level and I also open the heap file in order to add the necessary information of the first block. So, I create the first block of the heap file and I place the string: "This is a Heap File."

3.) HP_OpenFile(const char* filename, int* fileDesc)

In this function I open the heap file and I also check if the first block has the special information using strcmp.

4.) HP_CloseFile(int fileDesc)

In this function I simply close the heap file.

5.) HP_InsertEntry(int fileDesc, Record record)

In this function I insert the entries at each block using static variables to keep track of which block we're on and what slot the entry needs to placed on. If the slot variable is equal to 0 this means we have to allocate a new block to place the first entry. Otherwise we get an existing block. Memcpy function is used to place the entry inside the block as well as pointer arithmetics to get which slot it needs to placed on. Also, every time a record is placed, a "bytes" variable is incremented by the size of the record. If its value exceeds the size of the block this means no more records can placed so we have to allocate a new block for the next entry.

6.) HP_PrintAllEntries(int fileDesc, char* attrName, void* value)

In this function the same values as in HP_InsertEntry are used to keep track of which block we're on and what slot the record is on. We get how many blocks are in the heap file and for every block we get each record. We print each record, if the argument "value" is NULL, otherwise we check the field name and its corresponding value to see whether it will be printed or not.

7.) HP_GetEntry(int fileDesc, int rowId, Record* record)

In this function we get how many records are in each block, and we use div to get which block the rowId is on and mod to get at which slot the rowId is placed in the block. After that we use pointer arithmetics and casting to get the record we need and we store it in the "record" argument.

I didn’t implement anything regarding the Bonus question.