CSCI 201: Fundamentals of Computer Science  
Instructor: Pranava K. Jha

Some simple exercises on linked lists

In all exercises below, a node has the following blueprint:

```c
struct nodeType
{
    int info;
    nodeType* link;
};
typedef nodeType* nodeTypePtr;
```

1. Present the output in its content and form when the following piece of code is executed.

```c
nodeType* list;
nodeType* ptr;

list = new nodeType;
list->info = 20;
ptr = new nodeType;
ptr->info = 28;
ptr->link = NULL;
list->link = ptr;
ptr = new nodeType;
ptr->info = 30;
ptr->link = list;
list = ptr;
ptr = new nodeType;
ptr->info = 42;
ptr->link = list->link;
list->link = ptr;
ptr = list;
while(ptr != NULL)
{
    cout << ptr->info << endl;
    ptr = ptr->link;
}
```
2. A linked list appears below. H, A and B are of the type `nodePtr`.

```
    H  ->  18  ->  32  ->  23  ->  16  ->  43  ->  87
      |     |     |     |     |     |   \\
    A   |     |     |     |     |   \  \\
          |     |     |     |     |   B
```

a) Present the output of each of the following C++ statements.
   - `cout << H->info;`
   - `cout << A->info;`
   - `cout << B->link->info;`
   - `cout << H->link->link->info;`

b) Write statements to do the following.
   - Make A point to the node containing info 23.
   - Make H point to the node containing info 16.
   - Make B point to the last node in the list.
   - Set the value of the node containing 43 to 35.
   - Make H point to an empty list.

c) What is the output of the following piece of code?

```cpp
nodePtr p = H;
while (p != NULL)
{
    cout << p->info << " ";
    p = p->link;
}
cout << endl;
```