Task 1:

#include <iostream>

#include <iomanip>

using namespace std;

struct ListRec {

char value;

ListRec\* next;

};

int main() {

ListRec\* head;

head = nullptr;

head = new ListRec;

head->value = 'A';

head->next = new ListRec;

head->next->value = 'C';

head->next->next = new ListRec;

head->next->next->value = 'M';

head->next->next->next = nullptr;

return 0;

}

Task 2:

void print(ListRec\* listHead) {

while (listHead != nullptr) {

cout << listHead->value << " ";

listHead = listHead->next;

}

cout << endl;

}

Task 3:

void deepCopy(ListRec\* oldListHead, ListRec\* &newListHead) {

ListRec\* oldHead = oldListHead;

ListRec\* newHead = nullptr;

if (oldListHead == nullptr) {

return;

}

else {

while (oldHead != nullptr) {

if (newListHead == nullptr) {

newListHead = new ListRec;

newListHead->value = oldHead->value;

newHead = newListHead;

oldHead = oldHead->next;

}

else {

newHead->next = new ListRec;

newHead->next->value = oldHead->value;

newHead = newHead->next;

oldHead = oldHead->next;

}

}

newHead->next = nullptr;

}

}

Task 4:

#include <iostream>

#include <iomanip>

#include <cmath>

using namespace std;

struct ListRec {

char value;

ListRec\* next;

};

void print(ListRec\* listHead);

void deepCopy(ListRec\* oldListHead, ListRec\* &newListHead);

int main() {

ListRec\* head, \*headCopy;

head = nullptr;

headCopy = nullptr;

head = new ListRec;

head->value = 'A';

head->next = new ListRec;

head->next->value = 'C';

head->next->next = new ListRec;

head->next->next->value = 'M';

head->next->next->next = nullptr;

print(head);

deepCopy(head, headCopy);

cout << "The new list is: ";

print(headCopy);

headCopy->value = 'J';

cout << "The new list is: ";

print(headCopy);

return 0;

}

void print(ListRec\* listHead) {

while (listHead != nullptr) {

cout << listHead->value << " ";

listHead = listHead->next;

}

cout << endl;

}

//

void deepCopy(ListRec\* oldListHead, ListRec\* &newListHead) {

ListRec\* oldHead = oldListHead;

ListRec\* newHead = nullptr;

//newListHead = nullptr;

if (oldListHead == nullptr) {

return;

}

else {

while (oldHead != nullptr) {

if (newListHead == nullptr) {

newListHead = new ListRec;

newListHead->value = oldHead->value;

newHead = newListHead;

oldHead = oldHead->next;

}

else {

newHead->next = new ListRec;

newHead->next->value = oldHead->value;

newHead = newHead->next;

oldHead = oldHead->next;

}

}

newHead->next = nullptr;

}

}

