Week 2

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int main(void) {

struct bankEmployee {

char name[20];

int salary;

struct bankEmployee \*next;

};

typedef struct bankEmployee BANKEmployee;

typedef BANKEmployee \*BANKEmployeePtr;

/\* Question 1 a \*/

BANKEmployeePtr startPtr = NULL;

BANKEmployeePtr newPtr;

BANKEmployeePtr prevPtr;

BANKEmployeePtr crntPtr;

BANKEmployeePtr tempPtr;

/\* Question 1 b \*/

/\* Justin \*/

newPtr = (BANKEmployee\*) malloc(sizeof(BANKEmployee));

strcpy(newPtr->name, "Justin");

newPtr->salary = 1000;

newPtr->next = NULL;

startPtr=newPtr;

/\*

\* 0x10\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\* / name |salary| next \

\* |Justin| 1000 | NULL |

\* \\_\_\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_\_\_/

\* ^

\* /|\

\* |

\* |

\* startPtr

\* newPtr

\* both point to the struct itself

\*

\*/

/\* Question 1 c \*/

/\* Sam \*/

newPtr = (BANKEmployee\*) malloc(sizeof(BANKEmployee));

strcpy(newPtr->name, "Sam");

newPtr->salary = 999;

newPtr->next = NULL;

startPtr->next = newPtr;

/\*

\* Adding Sam

\*

\* 0x10\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\* / name |salary| next \

\* |Justin| 1000 | 0x20 | startPtr points to this struct

\* \\_\_\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_\_\_/

\* |

\* \_\_\_\_\_\_\_\_\_\_\_\_\_/

\* /

\* \/

\* 0x20\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\* / name |salary| next \

\* | Sam | 999 | NULL | newPtr points to this struct

\* \\_\_\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_\_\_/

\*

\*/

/\* Antony \*/

newPtr = (BANKEmployee\*) malloc(sizeof(BANKEmployee));

strcpy(newPtr->name, "Antony");

newPtr->salary = 200;

crntPtr=startPtr;

newPtr->next = crntPtr;

startPtr = newPtr;

/\* Tony \*/

newPtr = (BANKEmployee\*) malloc(sizeof(BANKEmployee));

strcpy(newPtr->name, "Justin");

newPtr->salary = 300;

prevPtr = crntPtr->next;

prevPtr->next = newPtr;

newPtr->next = NULL;

/\*\*/

/\* Peter \*/

newPtr = (BANKEmployee\*) malloc(sizeof(BANKEmployee));

strcpy(newPtr->name, "Peter");

newPtr->salary = 400;

crntPtr = prevPtr;

prevPtr = startPtr->next;

prevPtr->next = newPtr;

newPtr->next = crntPtr;

/\*\*/

/\*

\* Adding Antony, Tony and Peter.

\*

\*

\* startPtr == 0x30

\* newPtr == 0x50

\* crntPtr == 0x20

\* prevPtr == 0x10

\*

\* 0x30\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\* / name |salary| next \

\* |Antony| 200 | 0x10 |

\* \\_\_\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_\_\_/

\*

\*

\*

\* 0x10\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\* / name |salary| next \

\* |Justin| 1000 | 0x50 |

\* \\_\_\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_\_\_/

\*

\*

\*

\* 0x50\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\* / name |salary| next \

\* | Peter| 400 | 0x20 |

\* \\_\_\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_\_\_/

\*

\*

\*

\* 0x20\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\* / name |salary| next \

\* | Sam | 999 | 0x40 |

\* \\_\_\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_\_\_/

\*

\*

\*

\* 0x40\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\* / name |salary| next \

\* | Tony | 300 | NULL |

\* \\_\_\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_\_\_/

\*

\*/

/\* Question 1 d \*/

printf("\n");

crntPtr = startPtr;

while(crntPtr != NULL) {

printf("Name:\t%s\n"

"Salary:\t%d\n\n",

crntPtr->name,

crntPtr->salary);

crntPtr = crntPtr->next;

}

/\* Question 1 e \*/

while(startPtr != NULL) {

crntPtr = startPtr->next;

free(startPtr);

startPtr = crntPtr;

}

/\* Question 2 \*/

struct studentname {

char letter;

struct studentname \*next;

};

typedef struct studentname STUDENTName;

typedef STUDENTName \*STUDENTNamePtr;

/\* Create the pointers for navigating the list. \*/

STUDENTNamePtr newNode;

STUDENTNamePtr head;

/\* Add items to list in alphabetical order. \*/

newNode = (STUDENTName\*) malloc(sizeof(STUDENTName));

newNode->letter = 'u';

newNode->next = NULL;

head = newNode;

newNode = (STUDENTName\*) malloc(sizeof(STUDENTName));

newNode->letter = 'm';

newNode->next = head;

head = newNode;

newNode = (STUDENTName\*) malloc(sizeof(STUDENTName));

newNode->letter = 'l';

newNode->next = head;

head = newNode;

newNode = (STUDENTName\*) malloc(sizeof(STUDENTName));

newNode->letter = 'H';

newNode->next = head;

head = newNode;

newNode = (STUDENTName\*) malloc(sizeof(STUDENTName));

newNode->letter = 'e';

newNode->next = head;

head = newNode;

/\* Print the list. \*/

STUDENTNamePtr currentNode = head;

while(currentNode != NULL) {

printf("%c ", currentNode->letter);

currentNode = currentNode->next;

}

printf("\n");

while(head != NULL) {

currentNode = head->next;

free(head);

head = currentNode;

}

return 0;

}

