Sabancı University

Faculty of Engineering and Natural Sciences

CS204 Advanced Programming

Spring 2024

Homework 3 – Employee-Project Management System using a Hybrid Data Structure

Due: 03/04/2024, Wednesday, 21:00

PLEASE NOTE:

Your program should be a robust one such that you have to consider all relevant programmer mistakes and extreme cases; you are expected to take actions accordingly!

You can NOT collaborate with your friends and discuss solutions. You have to write down the code on your own. Plagiarism and homework trading will not be tolerated!

Introduction

In this homework, you are asked to implement a two-dimensional hybrid linked list (2DHLL) structure that will be used to store information about employees and their assigned projects. Along with the 2DHLL structure, you are also asked to implement and use a stack structure to keep track of the changes that are done on the 2DHLL structure for possible undoings.

The main program displays a menu offering some options to the user to interact with both data structures. According to the choice that the user enters, the program acts on and performs the relevant changes on both of the data structures.

The main program, which utilizes two different classes, is given to you in the homework package. What you will do is to design and implement these two classes, one for 2DHLL structure and one for the stack.

Program Flow

The program starts by displaying a menu with a predefined set of options. The user enters an option number from the displayed menu. In case of an invalid option number entered by the user, the program will output a message indicating that, and then wait for another input from the user. Once a valid input (option number) is entered, the program will either ask for more information that is related to the chosen option and then perform the actions related to this option, or if there is no extra information needed for the chosen option, the program will perform the actions related to this option directly.

The table below shows the information related to menu options.

Option number and title	**Description and expected action to be taken**	**Needed input from the user**
1. 'is_assigned_to'	Assigning an employee to a project.	- Employee name - Project name - Project priority
2. 'is_withdrawn_from'	Withdrawing an employee from a project	- Employee name - Project name
3. 'print_the_entire_list'	Prints the entire 2DHLL in a predefined format (see the sample runs)	
4. 'print_employee_project'	Prints the projects of a specific employee in the order of their priorities (ascending or descending)	- Employee name - display order (1 for ascending order; 0 for descending order)
5. 'undo	Undoes the last action that affected the 2DHLL	
6. 'exit'	Exits the program	

The program runs until menu option 6 will be entered. After 6 is entered, all dynamically allocated memory is deallocated and the program finishes.

Employee and project names are assumed to be all lowercase letters. Moreover, they are single words. Moreover, project priorities are all entered and used as positive integer values.

The implementation of the main program flow is given to you as a cpp file. You will use this without changing even a single character. Your task is to implement two classes that are explained below..

Data Structures and Classes

In your program, you will use two types of data structures each implemented as a separate class.

1. Hybrid 2DHLL Data Structure and the Corresponding Class

The first data structure, which is the main one, is for the 2DHLL data structure. It is the one that will be used to store information about the employees and their assigned projects.

Figure 1 shows a general structure for the 2DHLL data structure. As you can see from the figure, there are two types of nodes in this data structure, namely: *`EmployeeNode`* and *`ProjectNode`*. These two nodes are further explained in detail in the 1.1 Nodes Structures subsection below.

2DHLL is structured such that each row stores the information of an employee and his/her assigned projects. Specifically, each row starts with a node of type *`EmployeeNode`*, and the rest of the row contains one or more nodes of type *`ProjectNode`.* Each employee in this structure should have at least one project assigned.

For an employee, his or her assigned projects should be organized in ascending order of the projects' priorities. However, there must not be two projects with the same priority in a particular row. The employees, on the other hand, are organized in ascending alphabetical order of their names.

The name of the class is `EmployeeProject2DLL`.

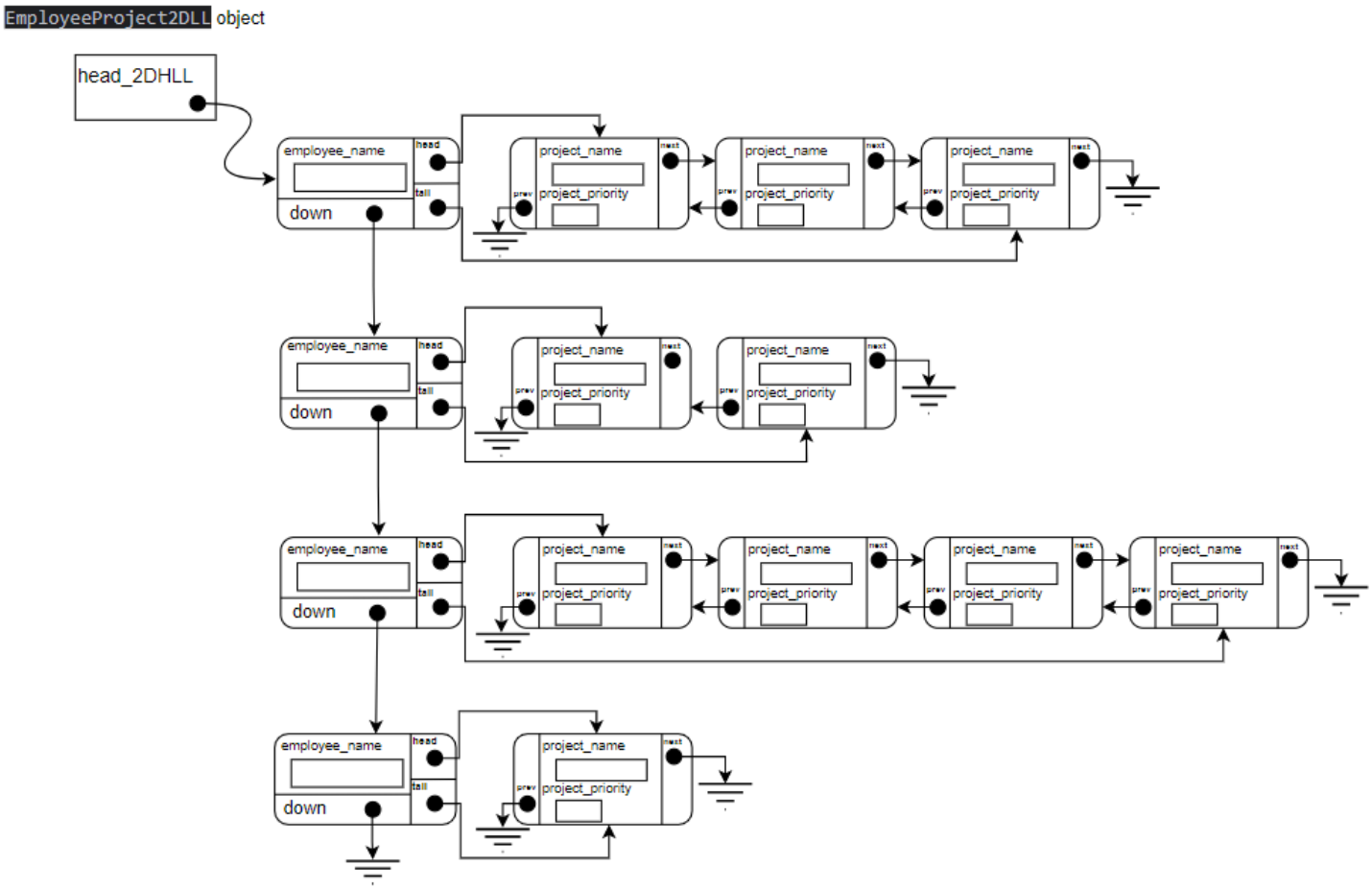

Figure 1. A general structure for the 2DHLL data. This is a generic view of the structure. In the program, there could be different rows and columns.

1.1 Nodes Structures

Below are the structs for the two types of nodes in the 2DHLL data structure and `EmployeeProject2DLL` class.

```
struct ProjectNode
{
    string project_name;
    int project_priority;
    ProjectNode * next;
    ProjectNode * prev;
};
```

```
struct EmployeeNode
{
    string employee_name;
    ProjectNode * head;
    ProjectNode * tail;
    EmployeeNode * down;
};
```

`EmployeeNode`

It is the node responsible for storing information of an employee. It stores the name of the employee, and three pointers. The projects assigned to this employee are stored in a doubly linked list. This is the row of 2DHLL and each row's `head` and `tail` pointers are used to point to the first (lowest priority project) and the last (highest priority project) nodes of the corresponding row, respectively. The `down` pointer is used to create a regular linked list among the `EmployeeNode`s.

`ProjectNode`

It is the node responsible for storing the information of a project. It stores the name of the project, its priority, and two pointers. These two pointers are connecting the project nodes in a two-way manner in order to create a doubly-linked list.

For both node structs, you may add constructors.

1.2 Member Functions:

Below we explain the list of the member functions of the `EmployeeProject2DLL` class that are needed for the program to run successfully using the provided main function flow. Feel free to implement other private member functions for the sake of modularity (to be used in the implementation of the required public member functions), but you are not allowed to implement extra public ones.

- **Default constructor**: Creates an empty object with no employee.

- **isEmployeeAssignedToProject**: This function takes two parameters, an employee name and a project name. It returns true/false depending on whether the employee is assigned to the project or not.

- **updateProjectPriority**: This function takes three parameters; an employee name, a project name, and a project priority. The last parameter (project priority) is a reference parameter. This function assumes that the project has already been assigned to the employee and the aim of the function is to update its priority. There are three outcomes of this function:
    - It does not change anything and returns false, if there is another project in the list of projects of that employee with the same priority as the parameter priority; in this case, the function should also display a message saying "The project priority has not been updated because there is another project with the same priority.".
    - If the existing assignment's priority is the same as the parameter priority, again the function returns false and does not change anything. This time, a message saying "The project priority is already the same as the new priority." is displayed.
    - Otherwise, it updates the priority of the project of that employee as the parameter priority, reorders the list (if needed) and then returns true. Moreover, the previous priority must be returned as the reference parameter.

- **assignEmployeeToProject**: This function takes three parameters; an employee name, a project name and a project priority. The aim of this function is to assign the project to the employee, if possible. If the project cannot be added to the employee's project list due to having another project with the same priority, then the structure does not change and the function returns false (in this case, the function should also display a message saying "The project has not been added because there is another project with the same priority."). Otherwise, the function returns true and assigns the project to the employee by observing the ordering rules and if needed by adding a new employee row.

- **withdrawEmployeeFromProject**: This function takes three parameters, an employee name, a project name and project priority (the last one is a reference parameter). This function assumes that there exists such an employee-project pair in the data structure and priority is not a criterion for selection. The function simply deletes the project node from the employee list and returns its priority as the reference parameter. The function itself does not return and display anything. Be aware that you cannot have employees with no projects assigned. If the project to be deleted is the only project that employee has, then the employee node has to be deleted as well.

- **printTheEntireList**: This function prints the content of the entire 2DHLL structure in a predefined format (employees in ascending alphabetical order; projects per employee are in ascending priority order). If the list is empty, it prints a specific message ("The list is empty.") indicating that. Please refer to sample runs for the output format~~, and specific message~~.

- **printEmployeeProjects**: This function takes two parameters, an employee name, and an integer value indicating whether to print the projects of that employee in an ascending (1) or descending (0) order according to the priorities of the projects. Please see the sample runs for the display format. If the list is totally empty, then a specific message is displayed ("There are no employees in the list."). If the list is not empty but there is no such employee, then the message to be displayed is "The employee is not in the list.".

- **undo:** This function takes four parameters: (i) a char type indicating the type of the operation, (ii) an employee name, (iii) a project name, and (iv) project priority. It recognizes the operation type and based on that, it performs an action that reverses that operation. There are three types of operations: 'a' for assigning a project; 'w' for withdrawing a project; 'u' for updating the priority of an existing project. This function should also display a message indicating the type of undo operation. These messages are "Undoing the assignment of a project." or "Undoing the withdrawal of a project." or "Undoing the update of a project priority.", depending of the operation.

- **clear**: This function takes no parameters, and is responsible for deallocating the dynamically-allocated memory of the entire 2DHLL structure.

2. Stack Class

The other class that you will design and implement is a linked-list based (i.e. dynamic) stack class. It is going to be used to record and manage the actions that are affecting the nodes of the 2DHLL data structure. In the main program, first an empty stack object is created. After that it is utilized through its `push` and `pop` functions for undo operations. The stack object is cleared at the end.

The name of the stack class is `UndoStack` and the class' only private data member must be the top pointer (no other data members). If you do not follow this rule, your homework will not be graded.

2.1 Node Structure

The `StackNode` is the node of the dynamic linked-list based stack structure. The fields are given below.

```
struct StackNode
{
    char operation;
    string employee_name;
    string project_name;
    int project_priority;
    StackNode * next;
};
```

Every node is used to record an action that has been performed on the main 2DHLL data structure (i.e. `EmployeeProject2DLL` object). The field `next` is used to link the nodes. The field `operation` is used to store the type of the operation (mentioned above in the undo function of `EmployeeProject2DLL` class). Other fields contain information regarding the operation.

You may write constructors for this struct.

2.2 Member Functions:

Below we explain the member functions of the UndoStack class that are needed for the main program to run successfully. Again, you can add extra private member functions but no extra public member functions. This is a strict rule to qualify for grading.

- **Default constructor**: Creates an empty stack.

- **push**: This function takes four parameters.
  (i) a char type parameter representing the operation, (ii) a string for an employee name, (iii) s string for a project name, and (iv) an integer for a project priority. It creates a stack node with the values of the function parameters and pushes it to the top of the stack.

- **pop**: This function takes four reference parameters.
  (i) a char type parameter representing the operation, (ii) a string for an employee name, (iii) a string for a project name, and (iv) an integer for a project priority. This function pops the top node of the stack and returns its data fields as reference parameters.

- **isEmpty**: This function returns true if the stack is empty; returns false otherwise.

- **clear**: This function takes no parameters, and is responsible for deallocating the dynamically-allocated memory of the stack

No other multiple data containers (built-in array, dynamic array, vector, matrix, 2D array, etc.) can be used.

Provided main cpp file and the requested class implementations

We have provided the main function of this homework to you within the homework pack. You have to use it directly and **without any modifications.**

As you see from the provided main function, the construction of the data structure via calling class member functions and operations are all implemented by calling the relevant member functions. What you have to do is to write the classes' header and implementation files so that they work with the given main function. That means, you will write two header and two cpp files; one pair for the `EmployeeProject2DLL` class and one pair for the `UndoStack` class.

The member functions used in main are self-explanatory and match with the operations. There also are comments in main. We expect you to explore the member function requirements by analyzing the given main program and the explanations in this homework document.

If you need more member functions, other than the ones that we use in main, you are more than welcome to write them, but as private member functions. Moreover, please first read the "Object Oriented Design Manifest" part below.

Object Oriented Design Manifest

We believe it is clear that you have to apply proper object oriented design principles in this homework, but our past experience says that most of the students are either unaware or not willing to follow these principles. Thus, we wanted to manifest some important rules about the class design and implementation.

1) According to the rule #1 of object oriented design, each member function **must be multi-purpose and must perform as single task as possible**. We picked the member functions used in main using this principle. You may add some extra member functions to be used internally for the implementations of the member functions (not in main since you are not allowed to change main), but please do not forget this principle while doing so.

2) Another important rule of object oriented programming is to hide the details of the class from the programmer that uses this class. Particularly for this homework, this means that in any free function and in the main function, we should not reach the head and tail of the doubly linked lists, and the main head of the hybrid data structure. We have to make all of the updates, deletes, searches, etc. in the main program through the member functions. Yes, technically it is possible and really easy to write a member function that returns heads and tails so that we can freely manipulate the data structure in the main program, but this is totally against the spirit behind object oriented programming and **we do not do this**. Actually, since you are not allowed to change main, we guarantee that you will not manipulate the data structure using heads and tail in main, but we wanted to make this clarification here.

3) There are two different classes in this homework and they are directly used in the main program. However, a particular class' member functions are not directly used in the other one's member functions. Thus, please do not use a member function of a class in the other one.

4) `UndoStack` class is the implementation of a stack data structure. You have to implement its member function using the stack data structure idea and rules.

5) Class and struct definitions will be in two separate header files (one for each class) of which the names are clear in the given main program. Do not mix/merge the classes' header files.

6) In two cpp files, you have to have the member function implementations of two classes separately. If you want to have some extra free functions to help the implementation of the member functions, also write them in this file (not in the main file). If you will write some helper free functions, please remark that the object oriented design rules #2 and #3 above are also valid for free functions.

7) You will **not** submit the provided main file. You will submit only `UndoStack` and `EmployeeProject2DLL` classes' header and cpp (implementation) files. Please see the submission guidelines below.

Some Important Programming Rules

Please do not use any non-ASCII characters (Turkish or other) in your code (not even as comments). And also do not use non-ASCII characters in your file and folder names. We really mean it; otherwise, you may encounter some errors.

In order to get a full credit, your programs must be efficient and well presented, presence of any redundant computation or bad indentation, or missing, irrelevant comments are going to decrease your grades. You also have to use understandable identifier names, informative introduction and prompts. Modularity is also important; you have to use functions wherever needed and appropriate. Since using classes is mandated in this homework, a proper object-oriented design and implementation will also be considered in grading.

Since you will use dynamic memory allocation in this homework, it is very crucial to properly manage the allocated area and return the deleted parts to the heap whenever appropriate. Inefficient use of memory may reduce your grade.

When we grade your homework, we pay attention to these issues. Moreover, in order to observe the real performance of your codes, we may run your programs in *Release* mode and **we may test your programs with very large test cases**. Of course, your program should work in *Debug* mode as well.

You are **not** allowed to use codes found somewhere online. These restrictions include code repositories, sites like geeksforgeeks, codes generated by GenAI tools, etc. Moreover, you are not allowed to use any statement, command, concept, topic that has not been covered in CS201 and CS204 (until now). Trying to find help online would generally cause such a problem. Thus, you always have to find help within the course material.

You cannot use break and continue statements. Global variables cannot be used either.

You are allowed to use sample codes shared with the class by the instructor and TAs. However, you cannot start with an existing .cpp or .h file directly and update it; you have to start with an empty file. Only the necessary parts of the shared code files can be used and these parts must be clearly marked in your homework by putting comments like the following. Even if you take a piece of code and update it slightly, you have to put a similar marking (by adding "`and updated`" to the comments below.

```
/* Begin: code taken from lab1.cpp */

…

/* End: code taken from lab1.cpp */
```

Since CodeRunner configuration in this homework does not allow to use extra non-standard C++ library/function/class files (other than the class files that you will implement), if you want to use such functions/classes covered in the classes (such as strutils), you will need to copy the necessary declarations and implementations to your files to be submitted by following the citation rules mentioned above.

Submission Rules (PLEASE READ SINCE SOME PARTS CHANGED)

It'd be a good idea to write your name and last name in the program (as a comment line of course). Do not use any Turkish characters anywhere in your code (not even in comment parts). For example, if your full name is "Satılmış Özbugsızkodyazaroğlu", then you must type it as follows:

// Satilmis Ozbugsizkodyazaroglu

We use CodeRunner in SUCourse+ for submission. No other way of submission is possible. Since the functionality part of the grading process will be automatic, you have to strictly follow these guidelines; otherwise we cannot grade your homework.

The advantage of CodeRunner is that you will be able to test your code against sample test cases. However, the output should be exact, but the textual differences between the correct output and yours can be highlighted (by pressing "show differences" button) on the interface.

The submission mechanism is a bit different than the other assignments, so please read on very carefully. You will submit four files: `EmployeeProject2DLL.cpp`, `EmployeeProject2DLL.h`, `UndoStack.cpp` and `UndoStack.h` files. **`EmployeeProject2DLL.cpp` file's content will be copied and pasted into the "Answer" area** as in other assignments. **However, you will upload `EmployeeProject2DLL.h`, `UndoStack.cpp` and `UndoStack.h` files as attachment** to your submission in the relevant assignment submission page on SUCourse+.

The file names that you will upload must definitely be **`EmployeeProject2DLL.h`, `UndoStack.cpp`** and **`UndoStack.h`**; otherwise it does not work. Moreover, you will not upload the provided main.cpp file; we have already put it there.

Even any tiny change in the output format will result in your grade being zero (0) for that particular test case, so please test your programs yourself, and against the sample runs that are available at the relevant assignment submission page on SUCourse+ (CodeRunner).

In the CodeRunner, there are some visible and invisible (hidden) test cases. You can test your code via CodeRunner against the sample runs (by pressing the "Precheck" button). You can precheck as much as you can; there is no penalty for multiple prechecks. This Precheck is a bit different than "Check" that you have used in previous assignments. After pressing "Precheck" you will be able to see the performance of your code **only** for the visible test cases; you will see a green check mark (for success) or a red cross (for failure) at the beginning of each visible test case. There will **not** be a feedback for the hidden test cases and there will **not** be any cumulative feedback at the end. Moreover, there will **not** be "show differences" button during Precheck.

You will be able to get a feedback about whether or not your code passed all test cases, including the hidden ones, only after you finish your attempt and complete the submission process (while previewing the submission). After the submission, "show differences" button will also be there for you to highlight the textual differences between your output and the expected one.

If you see a problem after you finish your attempt and complete the submission process, you will be able to re-attempt and make another submission. There is no penalty for re-attempts. However, you have to finish attempt and complete the submission process every time; there is no automatic submission on the deadline.

This process ensures that you will know whether or not your code has successfully passed all the test cases (visible and hidden) before finalizing your ultimate submission. However, we keep our rights to add more test cases in the grading process after the submission. **Thus, please make sure that you have read this documentation carefully and covered/tested all possible cases, even some other cases you may not have seen on CodeRunner or the sample runs.** Due to these reasons, **your final grade may conflict with what you have seen on**

CodeRunner. We will also **manually** check your code against some criteria, comments, indentations and so on; hence, please do not object to your grade based on the **CodeRunner** results, but rather, consider every detail on this documentation and in general homework rules.

We will consider your last submission in grading with no exceptions. Thus, please make sure that last submission is your final solution version. Also, we still do not suggest that you develop your solution on CodeRunner, but rather on your IDE on your computer.

Last, even if you cannot completely finish your homework, you can still submit.

Please see the syllabus for general homework grading issues.

Plagiarism

Plagiarism is checked by automated tools, and we are very capable of detecting such cases. Be careful with that. Exchange of abstract ideas are totally okay but once you start sharing the code with each other, it is very probable to get caught by plagiarism. So, do NOT share any part of your code to your friends by any means or you might be charged as well, although you have done your homework by yourself.

Homework is to be done personally and you have to submit your own work. **Cooperation will NOT be counted as an excuse.**

Our experience shows that code taken from online sources or generated by AI tools also show resemblance; thus if you try to get such help, you also may be charged with plagiarism with a person that you do not know.

In case of plagiarism, the rules written in the Syllabus apply.

Sample Runs

Sample runs are given below, but these are not comprehensive, therefore you must consider **all possible cases** to get a full mark. User inputs are shown in **bold**.

We configured CodeRunner to test these sample runs for you (as visible test cases). However, there also are some hidden test cases that would affect your grade. We will not disclose the hidden test cases before the grading has been completed.

We do **not** recommend you to copy and paste the prompts and messages from this document since some hidden control characters and non-standard characters might cause problems in CodeRunner.

Sample run 1

```
Hello! Welcome to the Employees-Projects Management System.
Below is the list of options that the system supports at the moment.
Please choose the number corresponding to the option,
and then enter the necessary information to execute the option.
1. is_assigned_to
2. is_withdrawn_from
```

```
3. print_the_entire_list
4. print_employee_projects
5. undo
6. exit
Please enter option number:
3
The list is empty.
Please enter option number:
5
The undo stack is empty.
Please enter option number:
3
The list is empty.
Please enter option number:
6
Deallocating the 2DHLL and the undo stack and terminating the program.
```

Sample run 2

```
Hello! Welcome to the Employees-Projects Management System.
Below is the list of options that the system supports at the moment.
Please choose the number corresponding to the option,
and then enter the necessary information to execute the option.
1. is_assigned_to
2. is_withdrawn_from
3. print_the_entire_list
4. print_employee_projects
5. undo
6. exit
Please enter option number:
1
Please enter the employee name:
cem
Please enter the project name:
ai
Please enter the project priority:
4
Please enter option number:
3
cem: (ai, 4)
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
0
Please enter the employee name:
linda
The employee is not in the list.
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
0
Please enter the employee name:
cem
(ai, 4)
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
1
Please enter the employee name:
```

cem
(ai, 4)
Please enter option number:
6
Deallocating the 2DHLL and the undo stack and terminating the program.

Sample run 3

Hello! Welcome to the Employees-Projects Management System.
Below is the list of options that the system supports at the moment.
Please choose the number corresponding to the option,
and then enter the necessary information to execute the option.
1. is_assigned_to
2. is_withdrawn_from
3. print_the_entire_list
4. print_employee_projects
5. undo
6. exit
Please enter option number:
5
The undo stack is empty.
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
0
Please enter the employee name:
hassan
There are no employees in the list.
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
1
Please enter the employee name:
hassan
There are no employees in the list.
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
4
Invalid input.
Please enter option number:
2
Please enter the employee name:
hassan
Please enter the project name:
web
Either the employee is not in the list or is in the list but is not assigned to that project.
Please enter option number:
3
The list is empty.
Please enter option number:
5
The undo stack is empty.
Please enter option number:
1
Please enter the employee name:
hassan
Please enter the project name:
web

```
Please enter the project priority:
2
Please enter option number:
3
hassan: (web, 2)
Please enter option number:
1
Please enter the employee name:
hassan
Please enter the project name:
ai
Please enter the project priority:
1
Please enter option number:
3
hassan: (ai, 1) (web, 2)
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
0
Please enter the employee name:
hassan
(web, 2) (ai, 1)
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
1
Please enter the employee name:
ali
The employee is not in the list.
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
1
Please enter the employee name:
hassan
(ai, 1) (web, 2)
Please enter option number:
6
Deallocating the 2DHLL and the undo stack and terminating the program.
```

Sample run 4

```
Hello! Welcome to the Employees-Projects Management System.
Below is the list of options that the system supports at the moment.
Please choose the number corresponding to the option,
and then enter the necessary information to execute the option.
1. is_assigned_to
2. is_withdrawn_from
3. print_the_entire_list
4. print_employee_projects
5. undo
6. exit
Please enter option number:
1
Please enter the employee name:
emre
Please enter the project name:
ai
Please enter the project priority:
14
Please enter option number:
```

3
emre: (ai, 14)
Please enter option number:
1
Please enter the employee name:
emre
Please enter the project name:
mob
Please enter the project priority:
5
Please enter option number:
3
emre: (mob, 5) (ai, 14)
Please enter option number:
1
Please enter the employee name:
emre
Please enter the project name:
ml
Please enter the project priority:
7
Please enter option number:
3
emre: (mob, 5) (ml, 7) (ai, 14)
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
0
Please enter the employee name:
emre
(ai, 14) (ml, 7) (mob, 5)
Please enter option number:
1
Please enter the employee name:
emre
Please enter the project name:
game
Please enter the project priority:
15
Please enter option number:
3
emre: (mob, 5) (ml, 7) (ai, 14) (game, 15)
Please enter option number:
2
Please enter the employee name:
emre
Please enter the project name:
art
Either the employee is not in the list or is in the list but is not assigned to that project.
Please enter option number:
2
Please enter the employee name:
emre
Please enter the project name:
mat
Either the employee is not in the list or is in the list but is not assigned to that project.
Please enter option number:
3
emre: (mob, 5) (ml, 7) (ai, 14) (game, 15)
Please enter option number:
2
Please enter the employee name:

```
emre
Please enter the project name:
ai
Please enter option number:
3
emre: (mob, 5) (ml, 7) (game, 15)
Please enter option number:
2
Please enter the employee name:
emre
Please enter the project name:
ml
Please enter option number:
3
emre: (mob, 5) (game, 15)
Please enter option number:
2
Please enter the employee name:
emre
Please enter the project name:
mob
Please enter option number:
3
emre: (game, 15)
Please enter option number:
2
Please enter the employee name:
emre
Please enter the project name:
ai
Either the employee is not in the list or is in the list but is not assigned to that project.
Please enter option number:
2
Please enter the employee name:
emre
Please enter the project name:
game
Please enter option number:
3
The list is empty.
Please enter option number:
6
Deallocating the 2DHLL and the undo stack and terminating the program.
```

Sample run 5

```
Hello! Welcome to the Employees-Projects Management System.
Below is the list of options that the system supports at the moment.
Please choose the number corresponding to the option,
and then enter the necessary information to execute the option.
1. is_assigned_to
2. is_withdrawn_from
3. print_the_entire_list
4. print_employee_projects
5. undo
6. exit
Please enter option number:
1
Please enter the employee name:
emre
```

```
Please enter the project name:
bio
Please enter the project priority:
5
Please enter option number:
3
emre: (bio, 5)
Please enter option number:
1
Please enter the employee name:
emre
Please enter the project name:
geo
Please enter the project priority:
2
Please enter option number:
3
emre: (geo, 2) (bio, 5)
Please enter option number:
5
Undoing the assignment of a project.
Please enter option number:
3
emre: (bio, 5)
Please enter option number:
1
Please enter the employee name:
emre
Please enter the project name:
ai
Please enter the project priority:
7
Please enter option number:
3
emre: (bio, 5) (ai, 7)
Please enter option number:
1
Please enter the employee name:
emre
Please enter the project name:
web
Please enter the project priority:
6
Please enter option number:
3
emre: (bio, 5) (web, 6) (ai, 7)
Please enter option number:
5
Undoing the assignment of a project.
Please enter option number:
3
emre: (bio, 5) (ai, 7)
Please enter option number:
2
Please enter the employee name:
emre
Please enter the project name:
web
Either the employee is not in the list or is in the list but is not assigned to that project.
Please enter option number:
2
Please enter the employee name:
```

emre
Please enter the project name:
bio
Please enter option number:
3
emre: (ai, 7)
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
3
emre: (bio, 5) (ai, 7)
Please enter option number:
2
Please enter the employee name:
emre
Please enter the project name:
ai
Please enter option number:
3
emre: (bio, 5)
Please enter option number:
2
Please enter the employee name:
emre
Please enter the project name:
bio
Please enter option number:
3
The list is empty.
Please enter option number:
6
Deallocating the 2DHLL and the undo stack and terminating the program.

Sample run 6

Hello! Welcome to the Employees-Projects Management System.
Below is the list of options that the system supports at the moment.
Please choose the number corresponding to the option,
and then enter the necessary information to execute the option.
1. is_assigned_to
2. is_withdrawn_from
3. print_the_entire_list
4. print_employee_projects
5. undo
6. exit
Please enter option number:
1
Please enter the employee name:
cem
Please enter the project name:
data
Please enter the project priority:
3
Please enter option number:
3
cem: (data, 3)
Please enter option number:
1
Please enter the employee name:

cem
Please enter the project name:
web
Please enter the project priority:
1
Please enter option number:
1
Please enter the employee name:
ada
Please enter the project name:
prog
Please enter the project priority:
1
Please enter option number:
3
ada: (prog, 1)
cem: (web, 1) (data, 3)
Please enter option number:
1
Please enter the employee name:
zain
Please enter the project name:
web
Please enter the project priority:
5
Please enter option number:
1
Please enter the employee name:
tarek
Please enter the project name:
air
Please enter the project priority:
13
Please enter option number:
3
ada: (prog, 1)
cem: (web, 1) (data, 3)
tarek: (air, 13)
zain: (web, 5)
Please enter option number:
2
Please enter the employee name:
tarek
Please enter the project name:
web
Either the employee is not in the list or is in the list but is not assigned to that project.
Please enter option number:
2
Please enter the employee name:
tarek
Please enter the project name:
air
Please enter option number:
3
ada: (prog, 1)
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
1
Please enter the employee name:
ada
Please enter the project name:

lang
Please enter the project priority:
2
Please enter option number:
3
ada: (prog, 1) (lang, 2)
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
5
Undoing the assignment of a project.
Please enter option number:
3
ada: (prog, 1)
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
1
Please enter the employee name:
ali
Please enter the project name:
ai
Please enter the project priority:
3
Please enter option number:
3
ada: (prog, 1)
ali: (ai, 3)
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
1
Please enter the employee name:
ali
Please enter the project name:
ml
Please enter the project priority:
2
Please enter option number:
3
ada: (prog, 1)
ali: (ml, 2) (ai, 3)
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
5
Undoing the assignment of a project.
Please enter option number:
3
ada: (prog, 1)
ali: (ai, 3)
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
2
Please enter the employee name:
ali
Please enter the project name:
ai
Please enter option number:
3
ada: (prog, 1)

```
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
3
ada: (prog, 1)
ali: (ai, 3)
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
2
Please enter the employee name:
zain
Please enter the project name:
web
Please enter option number:
3
ada: (prog, 1)
ali: (ai, 3)
cem: (web, 1) (data, 3)
Please enter option number:
2
Please enter the employee name:
ada
Please enter the project name:
prog
Please enter option number:
3
ali: (ai, 3)
cem: (web, 1) (data, 3)
Please enter option number:
2
Please enter the employee name:
cem
Please enter the project name:
data
Please enter option number:
3
ali: (ai, 3)
cem: (web, 1)
Please enter option number:
2
Please enter the employee name:
cem
Please enter the project name:
web
Please enter option number:
3
ali: (ai, 3)
Please enter option number:
2
Please enter the employee name:
ali
Please enter the project name:
ai
Please enter option number:
3
The list is empty.
Please enter option number:
4
```

```
Please enter 1 for ascending order, or 0 for descending order:
0
Please enter the employee name:
ali
There are no employees in the list.
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
3
ali: (ai, 3)
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
1
Please enter the employee name:
ali
(ai, 3)
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
3
ali: (ai, 3)
cem: (web, 1)
Please enter option number:
6
Deallocating the 2DHLL and the undo stack and terminating the program.
```

Sample Run 7

```
Hello! Welcome to the Employees-Projects Management System.
Below is the list of options that the system supports at the moment.
Please choose the number corresponding to the option,
and then enter the necessary information to execute the option.
1. is_assigned_to
2. is_withdrawn_from
3. print_the_entire_list
4. print_employee_projects
5. undo
6. exit
Please enter option number:
1
Please enter the employee name:
cem
Please enter the project name:
data
Please enter the project priority:
3
Please enter option number:
1
Please enter the employee name:
cem
Please enter the project name:
web
Please enter the project priority:
1
Please enter option number:
3
cem: (web, 1) (data, 3)
Please enter option number:
1
Please enter the employee name:
ada
Please enter the project name:
```

prog
Please enter the project priority:
1
Please enter option number:
1
Please enter the employee name:
zain
Please enter the project name:
web
Please enter the project priority:
5
Please enter option number:
3
ada: (prog, 1)
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
1
Please enter the employee name:
tarek
Please enter the project name:
air
Please enter the project priority:
13
Please enter option number:
3
ada: (prog, 1)
cem: (web, 1) (data, 3)
tarek: (air, 13)
zain: (web, 5)
Please enter option number:
5
Undoing the assignment of a project.
Please enter option number:
3
ada: (prog, 1)
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
1
Please enter the employee name:
tarek
Please enter the project name:
air
Please enter the project priority:
13
Please enter option number:
3
ada: (prog, 1)
cem: (web, 1) (data, 3)
tarek: (air, 13)
zain: (web, 5)
Please enter option number:
2
Please enter the employee name:
tarek
Please enter the project name:
air
Please enter option number:
2
Please enter the employee name:
ada
Please enter the project name:
prog
Please enter option number:
3
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
2
Please enter the employee name:

zain
Please enter the project name:
web
Please enter option number:
3
cem: (web, 1) (data, 3)
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
1
Please enter the employee name:
acun
Please enter the project name:
show
Please enter the project priority:
100
Please enter option number:
3
acun: (show, 100)
ada: (prog, 1)
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
1
Please enter the employee name:
zeynep
Please enter the project name:
data
Please enter the project priority:
3
Please enter option number:
3
acun: (show, 100)
ada: (prog, 1)
cem: (web, 1) (data, 3)
zain: (web, 5)
zeynep: (data, 3)
Please enter option number:
1
Please enter the employee name:
zeynep
Please enter the project name:
web
Please enter the project priority:
4
Please enter option number:
1
Please enter the employee name:
zeynep
Please enter the project name:
ai
Please enter the project priority:
4
The project has not been added because there is another project with the same priority.
Please enter option number:
3
acun: (show, 100)
ada: (prog, 1)
cem: (web, 1) (data, 3)
zain: (web, 5)
zeynep: (data, 3) (web, 4)
Please enter option number:
1
Please enter the employee name:
zeynep
Please enter the project name:
ai

```
Please enter the project priority:
2
Please enter option number:
3
acun: (show, 100)
ada: (prog, 1)
cem: (web, 1) (data, 3)
zain: (web, 5)
zeynep: (ai, 2) (data, 3) (web, 4)
Please enter option number:
2
Please enter the employee name:
zeynep
Please enter the project name:
data
Please enter option number:
3
acun: (show, 100)
ada: (prog, 1)
cem: (web, 1) (data, 3)
zain: (web, 5)
zeynep: (ai, 2) (web, 4)
Please enter option number:
2
Please enter the employee name:
zeynep
Please enter the project name:
web
Please enter option number:
3
acun: (show, 100)
ada: (prog, 1)
cem: (web, 1) (data, 3)
zain: (web, 5)
zeynep: (ai, 2)
Please enter option number:
2
Please enter the employee name:
zeynep
Please enter the project name:
ai
Please enter option number:
3
acun: (show, 100)
ada: (prog, 1)
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
2
Please enter the employee name:
ada
Please enter the project name:
prog
Please enter option number:
2
Please enter the employee name:
cem
Please enter the project name:
web
Please enter option number:
3
acun: (show, 100)
cem: (data, 3)
zain: (web, 5)
Please enter option number:
2
Please enter the employee name:
cem
Please enter the project name:
data
Please enter option number:
```

```
2
Please enter the employee name:
acun
Please enter the project name:
show
Please enter option number:
3
zain: (web, 5)
Please enter option number:
2
Please enter the employee name:
zain
Please enter the project name:
web
Please enter option number:
3
The list is empty.
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
3
acun: (show, 100)
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
3
acun: (show, 100)
ada: (prog, 1)
cem: (web, 1) (data, 3)
zain: (web, 5)
zeynep: (ai, 2) (web, 4)
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
5
Undoing the assignment of a project.
Please enter option number:
5
Undoing the assignment of a project.
Please enter option number:
5
Undoing the assignment of a project.
Please enter option number:
3
acun: (show, 100)
ada: (prog, 1)
cem: (web, 1) (data, 3)
zain: (web, 5)
Please enter option number:
5
Undoing the assignment of a project.
```

```
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
3
ada: (prog, 1)
cem: (web, 1) (data, 3)
tarek: (air, 13)
zain: (web, 5)
Please enter option number:
5
Undoing the assignment of a project.
Please enter option number:
5
Undoing the assignment of a project.
Please enter option number:
5
Undoing the assignment of a project.
Please enter option number:
5
Undoing the assignment of a project.
Please enter option number:
5
Undoing the assignment of a project.
Please enter option number:
3
The list is empty.
Please enter option number:
5
The undo stack is empty.
Please enter option number:
6
Deallocating the 2DHLL and the undo stack and terminating the program.
```

Sample Run 8

```
Hello! Welcome to the Employees-Projects Management System.
Below is the list of options that the system supports at the moment.
Please choose the number corresponding to the option,
and then enter the necessary information to execute the option.
1. is_assigned_to
2. is_withdrawn_from
3. print_the_entire_list
4. print_employee_projects
5. undo
6. exit
Please enter option number:
1
Please enter the employee name:
hossam
Please enter the project name:
data
Please enter the project priority:
1
Please enter option number:
3
hossam: (data, 1)
Please enter option number:
2
Please enter the employee name:
hossam
Please enter the project name:
data
```

```
Please enter option number:
3
The list is empty.
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
3
hossam: (data, 1)
Please enter option number:
1
Please enter the employee name:
hossam
Please enter the project name:
ai
Please enter the project priority:
4
Please enter option number:
3
hossam: (data, 1) (ai, 4)
Please enter option number:
1
Please enter the employee name:
ali
Please enter the project name:
ai
Please enter the project priority:
1
Please enter option number:
3
ali: (ai, 1)
hossam: (data, 1) (ai, 4)
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
0
Please enter the employee name:
hossam
(ai, 4) (data, 1)
Please enter option number:
2
Please enter the employee name:
cem
Please enter the project name:
hum
Either the employee is not in the list or is in the list but is not assigned to that project.
Please enter option number:
3
ali: (ai, 1)
hossam: (data, 1) (ai, 4)
Please enter option number:
1
Please enter the employee name:
hossam
Please enter the project name:
ai
Please enter the project priority:
3
Please enter option number:
3
ali: (ai, 1)
hossam: (data, 1) (ai, 3)
Please enter option number:
1
Please enter the employee name:
umut
```

```
Please enter the project name:
game
Please enter the project priority:
4
Please enter option number:
3
ali: (ai, 1)
hossam: (data, 1) (ai, 3)
umut: (game, 4)
Please enter option number:
1
Please enter the employee name:
umut
Please enter the project name:
ai
Please enter the project priority:
1
Please enter option number:
3
ali: (ai, 1)
hossam: (data, 1) (ai, 3)
umut: (ai, 1) (game, 4)
Please enter option number:
1
Please enter the employee name:
hossam
Please enter the project name:
data
Please enter the project priority:
3
The project priority has not been updated because there is another project with the same priority.
Please enter option number:
3
ali: (ai, 1)
hossam: (data, 1) (ai, 3)
umut: (ai, 1) (game, 4)
Please enter option number:
5
Undoing the assignment of a project.
Please enter option number:
3
ali: (ai, 1)
hossam: (data, 1) (ai, 3)
umut: (game, 4)
Please enter option number:
1
Please enter the employee name:
hossam
Please enter the project name:
data
Please enter the project priority:
4
Please enter option number:
3
ali: (ai, 1)
hossam: (ai, 3) (data, 4)
umut: (game, 4)
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
2
Invalid input.
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
1
```

```
Please enter the employee name:
hossam
(ai, 3) (data, 4)
Please enter option number:
1
Please enter the employee name:
ali
Please enter the project name:
ai
Please enter the project priority:
1
The project priority is already the same as the new priority.
Please enter option number:
3
ali: (ai, 1)
hossam: (ai, 3) (data, 4)
umut: (game, 4)
Please enter option number:
1
Please enter the employee name:
umut
Please enter the project name:
data
Please enter the project priority:
4
The project has not been added because there is another project with the same priority.
Please enter option number:
3
ali: (ai, 1)
hossam: (ai, 3) (data, 4)
umut: (game, 4)
Please enter option number:
1
Please enter the employee name:
umut
Please enter the project name:
data
Please enter the project priority:
19
Please enter option number:
3
ali: (ai, 1)
hossam: (ai, 3) (data, 4)
umut: (game, 4) (data, 19)
Please enter option number:
1
Please enter the employee name:
umut
Please enter the project name:
ai
Please enter the project priority:
18
Please enter option number:
3
ali: (ai, 1)
hossam: (ai, 3) (data, 4)
umut: (game, 4) (ai, 18) (data, 19)
Please enter option number:
2
Please enter the employee name:
ali
Please enter the project name:
ai
Please enter option number:
3
hossam: (ai, 3) (data, 4)
```

```
umut: (game, 4) (ai, 18) (data, 19)
Please enter option number:
1
Please enter the employee name:
hossam
Please enter the project name:
ai
Please enter the project priority:
5
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
1
Please enter the employee name:
hossam
(data, 4) (ai, 5)
Please enter option number:
3
hossam: (data, 4) (ai, 5)
umut: (game, 4) (ai, 18) (data, 19)
Please enter option number:
2
Please enter the employee name:
hossam
Please enter the project name:
ai
Please enter option number:
3
hossam: (data, 4)
umut: (game, 4) (ai, 18) (data, 19)
Please enter option number:
2
Please enter the employee name:
umut
Please enter the project name:
ai
Please enter option number:
3
hossam: (data, 4)
umut: (game, 4) (data, 19)
Please enter option number:
1
Please enter the employee name:
umut
Please enter the project name:
data
Please enter the project priority:
4
The project priority has not been updated because there is another project with the same priority.
Please enter option number:
3
hossam: (data, 4)
umut: (game, 4) (data, 19)
Please enter option number:
5
Undoing the withdrawal of a project.
Please enter option number:
3
hossam: (data, 4)
umut: (game, 4) (ai, 18) (data, 19)
Please enter option number:
1
Please enter the employee name:
umut
Please enter the project name:
ai
```

```
Please enter the project priority:
1
Please enter option number:
3
hossam: (data, 4)
umut: (ai, 1) (game, 4) (data, 19)
Please enter option number:
5
Undoing the update of a project priority.
Please enter option number:
3
hossam: (data, 4)
umut: (game, 4) (ai, 18) (data, 19)
Please enter option number:
1
Please enter the employee name:
hossam
Please enter the project name:
data
Please enter the project priority:
1
Please enter option number:
3
hossam: (data, 1)
umut: (game, 4) (ai, 18) (data, 19)
Please enter option number:
1
Please enter the employee name:
umut
Please enter the project name:
data
Please enter the project priority:
1
Please enter option number:
4
Please enter 1 for ascending order, or 0 for descending order:
0
Please enter the employee name:
umut
(ai, 18) (game, 4) (data, 1)
Please enter option number:
3
hossam: (data, 1)
umut: (data, 1) (game, 4) (ai, 18)
Please enter option number:
6
Deallocating the 2DHLL and the undo stack and terminating the program.
```