Practical 5

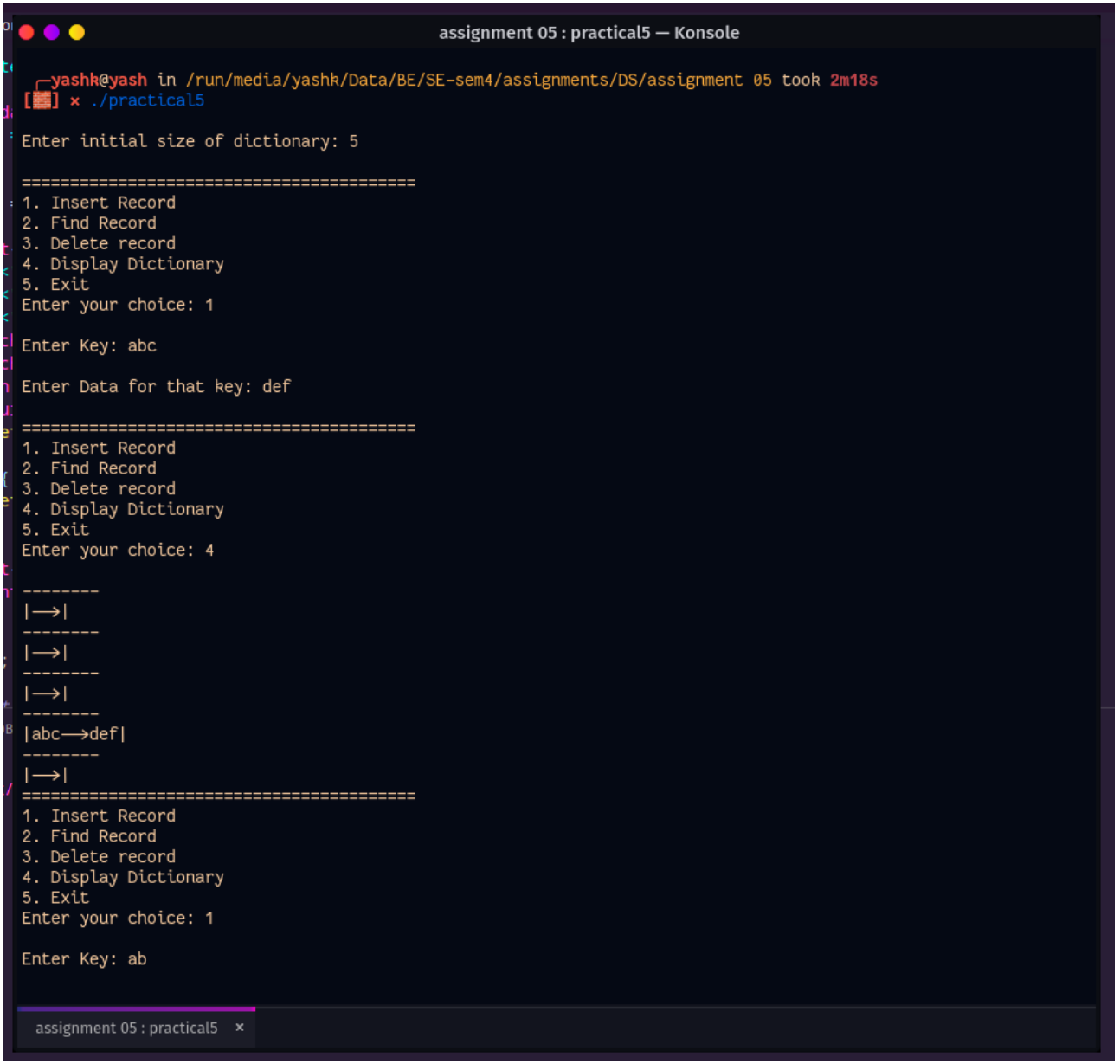
```
#include "iostream"
using namespace std;
struct node {
   string key;
   string data;
   node *next;
};
class Dictionary {
   node *hashmap;
   int size;
   public:
   Dictionary(int n) {
      hashmap = new node[n];
      this->size = n;
   }
   private:
   int generateHash(string key) {
      int hash = 0;
      for (int i = 0; i < key.length(); i++) {
         hash += int(key[i])*(i+1);
      }
      // return hash%this->size;
      return (key.length())%this->size;
   }
   void insert(string key, string data) {
      node *newRecord = new node();
      newRecord->key = key;
      newRecord->data = data;
      newRecord->next = NULL;
      int hash = generateHash(key);
      if(hashmap[hash].data == "") {
         hashmap[hash] = *newRecord;
      }
      else {
         node *current = &hashmap[hash];
         while(current != NULL) {
            if(current->key == key) {
               cout<<"\nRecord already exists";
               cout<<"\nCurrent Record: \n\t"<<current->key<<"--->"<<current->data<<endl;
               cout<<"\nDo you wanna update it (y/n)?";
               char ch;
               cin>>ch;
               if (ch == 'y' || ch == 'Y') {
                  current->data = data;
```

```
                return;
            }
            else {
                return;
            }
        }
        if(current->next != NULL) {
            current = current->next;
        }
        else {
            break;
        }
        // current = current->next;
    }
    current->next = newRecord;
  }
}
void search(string key) {
  int hash = generateHash(key);
  if(hashmap[hash].key == key) {
    cout<<"\nRecord Found";
    cout<<"\nIts Details: \n\t"<<hashmap[hash].key<<"--->"<<hashmap[hash].data<<endl;
  }
  else {
    node *current = &hashmap[hash];
    while(current != NULL) {
      if(current->key == key) {
        cout<<"\nRecord Found";
        cout<<"\nIts Details: \n\t"<<current->key<<"--->"<<current->data<<endl;
        return;
      }
      current = current->next;
    }
    cout<<"\nRecord Not Found";
  }
}
void deleteRecord(string key) {
  int hash = generateHash(key);

  node *current = &hashmap[hash];
  if(hashmap[hash].key == key) {
    if(current->next != NULL) {
      hashmap[hash] = *current->next;
    }
    else {
      node *empty = new node();
      hashmap[hash] = *empty;
    }
```

```
        }
        else {
            while(current != NULL) {
                if(current->next->key == key) {
                    node *temp = current->next;
                    current->next = current->next->next;
                    delete temp;
                    cout<<"\nRecord Deleted";
                    return;
                }
                current = current->next;
            }
            cout<<"\nRecord Not Found";
        }
    }
    void display() {
        for (int i = 0; i < this->size; i++) {
            node *current = &hashmap[i];
            cout<<"\n--------\n";
            while(current != NULL) {
                cout<<"|"<<current->key<<"-->"<<current->data<<"|\t";
                current = current->next;
            }
        }
    }
    void printMenu() {
        cout<<"\n=========================================";
        cout<<"\n1. Insert Record";
        cout<<"\n2. Find Record";
        cout<<"\n3. Delete record";
        cout<<"\n4. Display Dictionary";
        cout<<"\n5. Exit";
    }
public:
    void exec() {
        bool exit = false;
        while(!exit) {
            printMenu();
            cout<<"\nEnter your choice: ";
            int ch;
            cin>>ch;
            switch(ch) {
                case 1: {
                    string key, data;
                    cout<<"\nEnter Key: ";
                    cin>>key;
                    cout<<"\nEnter Data for that key: ";
                    cin>>data;
```

```
                insert(key, data);
                break;
            }
            case 2: {
                cout<<"\nEnter key you want to search: ";
                string key;
                cin>>key;
                search(key);
                break;
            }
            case 3: {
                cout<<"\nEnter key you want to Delete: ";
                string key;
                cin>>key;
                deleteRecord(key);
                break;
            }
            case 4:
                display();
                break;
            case 5:
                exit = true;
                break;
            default:
                cout<<"\nWrong choice";
        }
      }
   }
};
int main() {
   cout<<"\nEnter initial size of dictionary: ";
   int n;
   cin>>n;
   Dictionary *d = new Dictionary(n);
   d->exec();
   return 0;
}
```


assignment 05 : practical5 — Konsole
yashk@yash in /run/media/yashk/Data/BE/SE-sem4/assignments/DS/assignment 05 took 2m18s
[] × ./practical5
Enter initial size of dictionary: 5
==========
1. Insert Record
2. Find Record
3. Delete record
4. Display Dictionary
5. Exit
Enter your choice: 1
Enter Key: abc
Enter Data for that key: def
==========
1. Insert Record
2. Find Record
3. Delete record
4. Display Dictionary
5. Exit
Enter your choice: 4

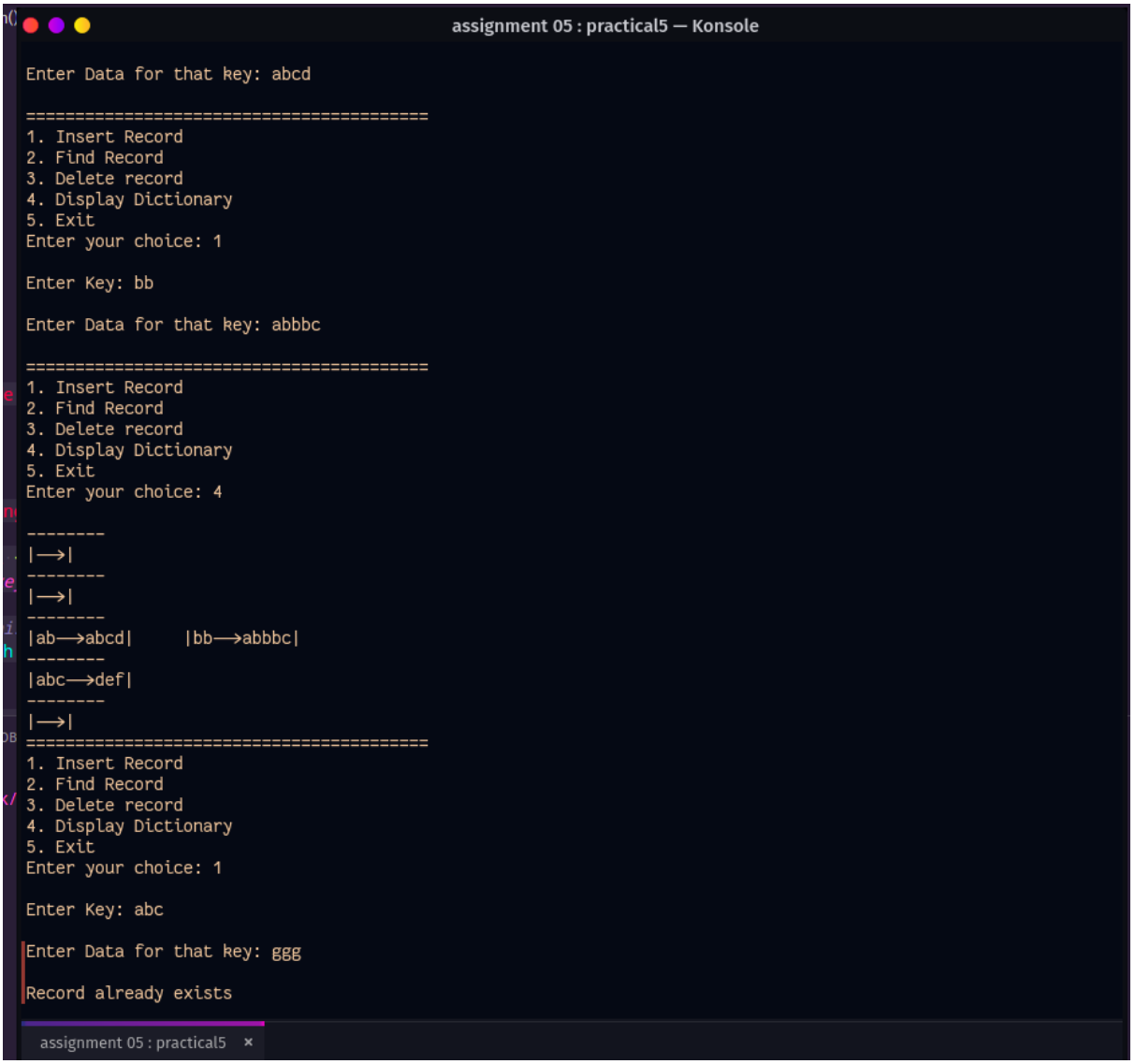
|⟶|

|⟶|

|⟶|

|abc⟶def|

|⟶|
==========
1. Insert Record
2. Find Record
3. Delete record
4. Display Dictionary
5. Exit
Enter your choice: 1
Enter Key: ab
assignment 05 : practical5 ×


assignment 05 : practical5 — Konsole
Enter Data for that key: abcd
==========
1. Insert Record
2. Find Record
3. Delete record
4. Display Dictionary
5. Exit
Enter your choice: 1
Enter Key: bb
Enter Data for that key: abbbc
==========
1. Insert Record
2. Find Record
3. Delete record
4. Display Dictionary
5. Exit
Enter your choice: 4

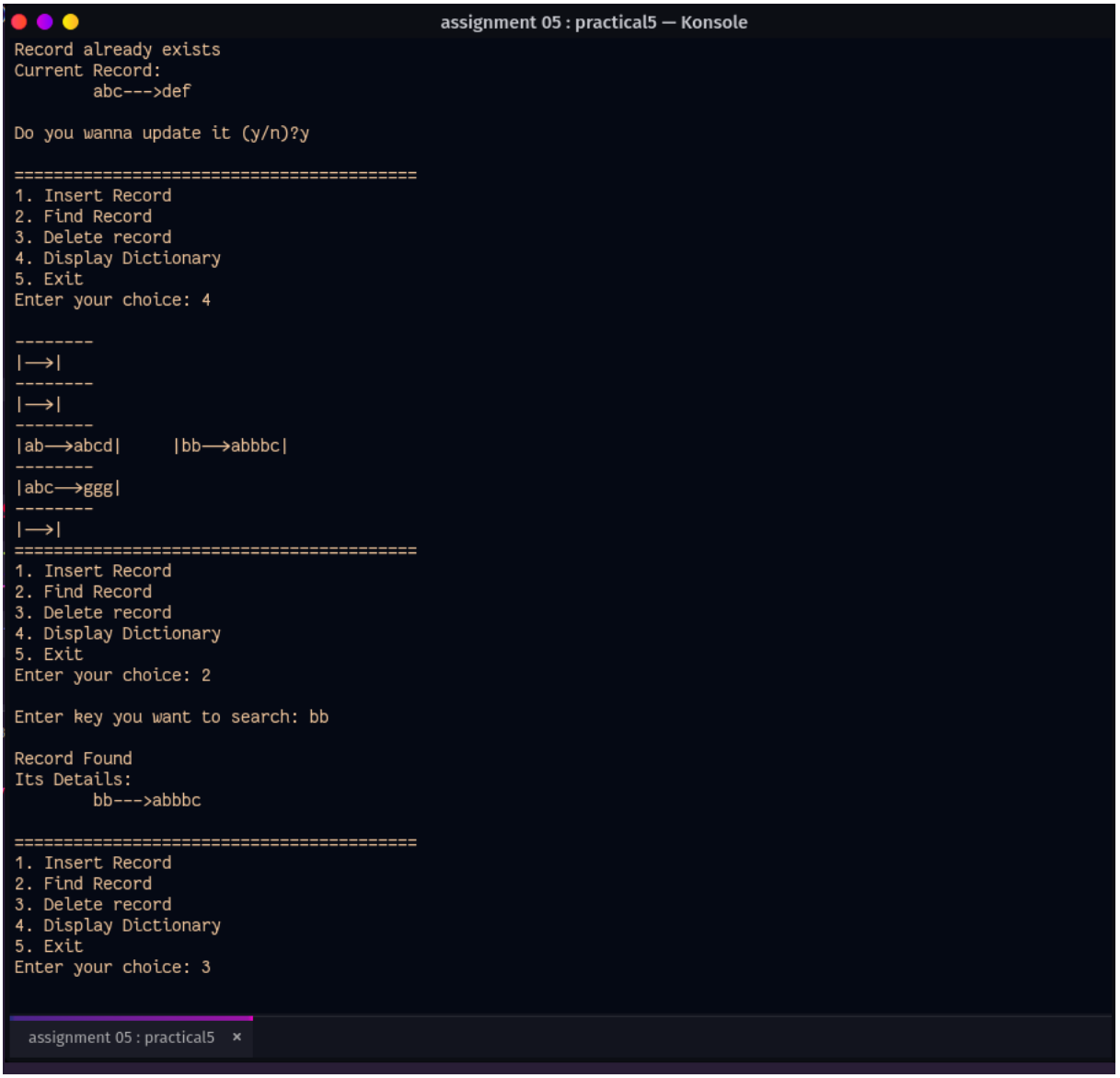
|⟶|

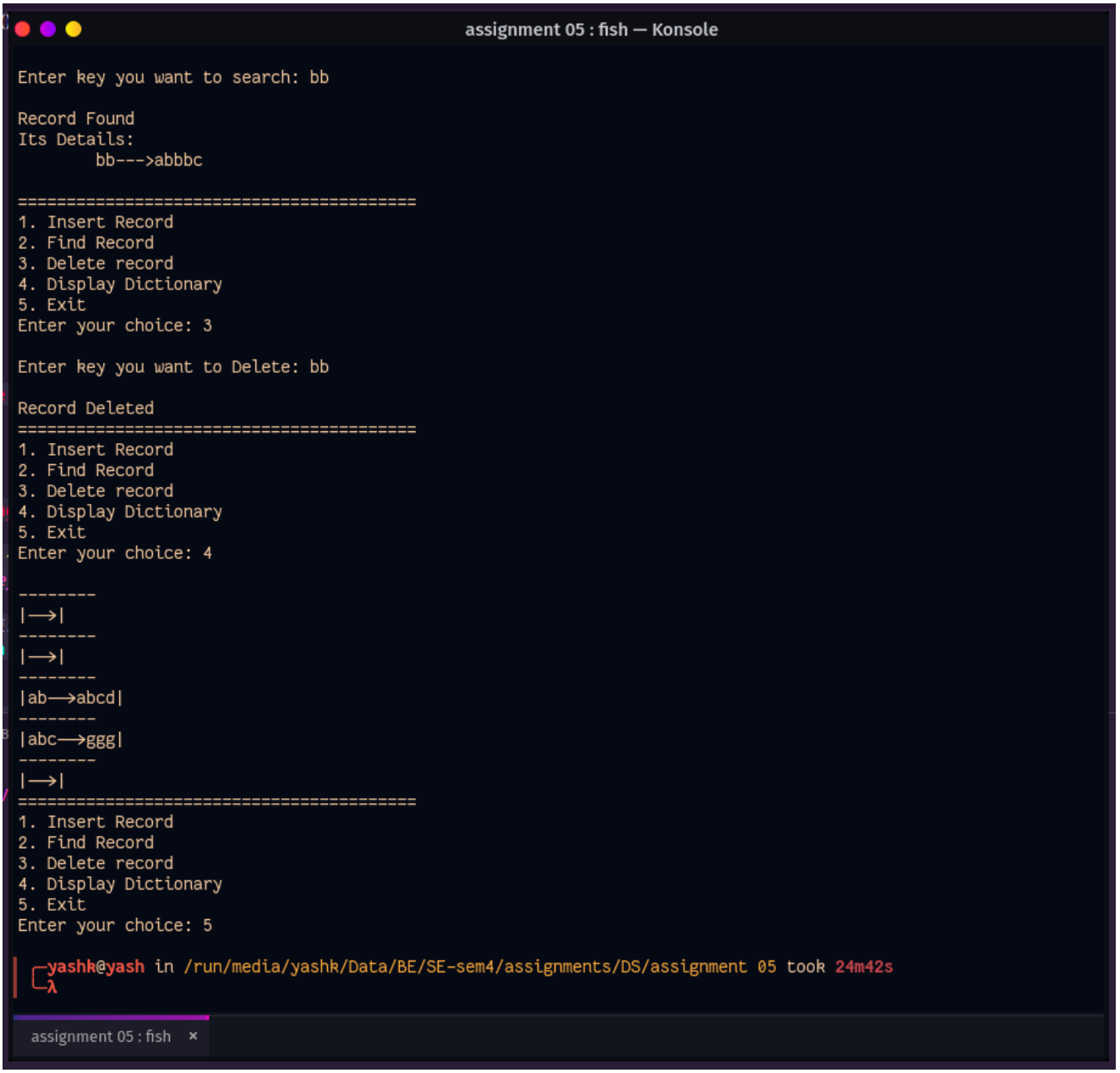
|⟶|

|ab⟶abcd| |bb⟶abbbc|

|abc⟶def|

|⟶|
==========
1. Insert Record
2. Find Record
3. Delete record
4. Display Dictionary
5. Exit
Enter your choice: 1
Enter Key: abc
Enter Data for that key: ggg
Record already exists
assignment 05 : practical5


assignment 05 : practical5 — Konsole
Record already exists
Current Record:
abc--->def
Do you wanna update it (y/n)?y
1. Insert Record
2. Find Record
3. Delete record
4. Display Dictionary
5. Exit
Enter your choice: 4
|ab⟶abcd|
|bb⟶abbbc|
|abc⟶ggg|
1. Insert Record
2. Find Record
3. Delete record
4. Display Dictionary
5. Exit
Enter your choice: 2
Enter key you want to search: bb
Record Found
Its Details:
bb--->abbbc
1. Insert Record
2. Find Record
3. Delete record
4. Display Dictionary
5. Exit
Enter your choice: 3
assignment 05 : practical5


assignment 05 : fish — Konsole
Enter key you want to search: bb
Record Found
Its Details:
bb--->abbbc
1. Insert Record
2. Find Record
3. Delete record
4. Display Dictionary
5. Exit
Enter your choice: 3
Enter key you want to Delete: bb
Record Deleted
1. Insert Record
2. Find Record
3. Delete record
4. Display Dictionary
5. Exit
Enter your choice: 4
|⟶|
|⟶|
|ab⟶abcd|
|abc⟶ggg|
|⟶|
1. Insert Record
2. Find Record
3. Delete record
4. Display Dictionary
5. Exit
Enter your choice: 5
yashk@yash in /run/media/yashk/Data/BE/SE-sem4/assignments/DS/assignment 05 took 24m42s
λ
assignment 05 : fish