**QUESTION 1**

Text

Description automatically generatedText

Description automatically generated

#include <iostream>

using *namespace* std;

*// Taking size of stack as 5*

#define SIZE 5

*template*<*class* T>

*class* Stack {

*// Taking data member top*

*int* top;

*// Initialising stack(array) of given size*

    T stack[SIZE]; *// size 5 declared globally*

*public:*

    Stack(){top = -1;}

*void* push(T *k*);

    T pop();

*int* topElement();

*bool* isFull();

*bool* isEmpty();

};

*template* <*class* T>

*int* Stack<T>::topElement(){

*// Initialising a variable to store top element*

*int* top\_element = stack[top];

*// Returning the top element*

    return top\_element;

}

*template* <*class* T>

*bool* Stack<T>::isEmpty(){ *// valid*

*// Till top in inside the stack*

    if (top == -1){return 1;}

    else{return 0;}

}

*template* <*class* T>

*bool* Stack<T>::isFull(){

*// Till top in inside the stack*

    if (top == (SIZE - 1)){return 1;}

    else{return 0;}

}

*template* <*class* T>

*void* Stack<T>::push(T *k*){ *//valid*

*// Checking whether stack is completely filled*

    if (isFull()) {

*// Display message when no elements can be pushed*

*// into it*

        cout << "Stack is full\n";

    }else{

*// Inserted element*

    cout << "Inserted element " << *k* << endl;

*// Incrementing the top by unity as element*

*// is to be inserted*

    top++;

*// Now, adding element to stack*

    stack[top] = *k*;

    }

}

*template* <*class* T>

T Stack<T>::pop(){ *// valid*

*// Initialising a variable to store popped element*

    T popped\_element = stack[top];

*// Decreasing the top as*

*// element is getting out from the stack*

    top--;

*// Returning the element/s that is/are popped*

    return popped\_element;

}

*void* Foo(Stack<*int*> *&X*){

    cout << "Enter Value" << endl;

*int* temp;

    cin >> temp;

*X*.push(temp);

}

*int* main(){

    string name = "Fahad Faruqi";

    cout << "--------------------------------------------------------------------------\n";;

    cout << "Name: " << name << endl;

    cout << "FILE: " << \_\_FILE\_\_ << "\tDATE: " << \_\_DATE\_\_ << endl;

    cout << "--------------------------------------------------------------------------\n";

*// Creating object of Stack class in main() method*

    Stack<*int*> i\_stack;

*// Adding elements to integer stack object*

*// Custom integer entries*

    i\_stack.push(2);

    i\_stack.push(54);

    i\_stack.push(255);

*// Now, removing element from integer stack*

    cout << i\_stack.pop() << " is removed from stack" << endl;

    Foo(i\_stack);

*// Print and display the top element in integer stack*

    cout << "Top element is " << i\_stack.topElement() << endl;

    return 0;

}

**OUTPUT**

PS C:\Users\Fahad\Desktop\212> cd "c:\Users\Fahad\Desktop\212\" ; if ($?) { g++ fahadFaruqiTakeHomeTest2Q1.cpp -o fahadFaruqiTakeHomeTest2Q1 } ; if ($?) { .\fahadFaruqiTakeHomeTest2Q1 }

--------------------------------------------------------------------------

Name: Fahad Faruqi

FILE: fahadFaruqiTakeHomeTest2Q1.cpp DATE: Dec 8 2022

--------------------------------------------------------------------------

Inserted element 2

Inserted element 54

Inserted element 255

255 is removed from stack

Enter Value

5

Inserted element 5

Top element is 5

Text

Description automatically generated

QUESTION 2 (unfinished unfortunately ☹)

#include <iostream>

#include <string>

using *namespace* std;

*struct* node{

    string Fruit; *//name*

*int* cnt; *//#fruits*

*public:*

    node (){

        cnt = 15;

    }

};

*int* hashFunction(string *key*){

*int* sum(0);

    for (*int* i(0); i < *key*.size(); i++){ *// adds up all char vals*

        sum += *int*(*key*[i]);

    }

*// cout << key <<"\t" << sum%11 << endl; //debug*

    return sum%11;

}

*int* main () {

    string name = "Fahad Faruqi";

    cout << "--------------------------------------------------------------------------\n";;

    cout << "Name: " << name << endl;

    cout << "FILE: " << \_\_FILE\_\_ << "\tDATE: " << \_\_DATE\_\_ << endl;

    cout << "--------------------------------------------------------------------------\n";

    node fruits[21];

    string fruitName[21] = {"Apple", "Banana", "Cherry", "Date", "Eldenberry", "Fig", "Guava", "Honeydew", "Jackfruit", "Kiwi", "Lemon", "Mango", "Nut", "Orange", "Pear",

    "Quince" "Raisin", "Strawberry", "Tangerine", "Watermelon", "Yumberry"};

    node \*hashArray[11][10];

*int* j = 0;

    for (*int* i = 0; i<21; i++){ *// initalize fruits*

        if (fruitName[i] == ""){ *// just incase ...*

            break;

        }

        fruits[i].cnt = i+1;

        fruits[i].Fruit = fruitName[i]; *// name set*

*// cout << i << endl;*

        while(hashArray[hashFunction(fruits[i].Fruit)][j] == NULL){ *// while loop checks that the index value and chain have space*

            j++;

        }

        hashArray[hashFunction(fruits[i].Fruit)][j] = &fruits[i]; *// after each fruit is initialized, the name gets turned into an index and the address of said fruit is stored at that index in the hashArray*

        j=0; *// reset*

    }

    cout << "\n\*\*\*\* PRINTED HASHTABLE \*\*\*\*\n";

    for(*int* i(0); i<11; i++){ *// goes through index*

        for(*int* j(0); j<10; j++){ *// goes through chain*

            if (hashArray[i][j-1] == NULL){break;}

            else {

                if(hashArray[i][j] != NULL){

                printf("Fruit: %s\tCnt: %i\n", hashArray[i][j]->Fruit.c\_str(), hashArray[i][j]->cnt);

                }

            }

            cout << i << j << endl;

        }

    }

    return 0;

}

Text

Description automatically generatedText

Description automatically generated

**OUTPUT**Graphical user interface

Description automatically generatedPS C:\Users\Fahad\Desktop\212> cd "c:\Users\Fahad\Desktop\212\" ; if ($?) { g++ test3.cpp -o test3 } ; if ($?) { .\test3 }

--------------------------------------------------------------------------

Name: Fahad Faruqi

FILE: test3.cpp DATE: Dec 8 2022

--------------------------------------------------------------------------

\*\*\*\* PRINTED HASHTABLE \*\*\*\*

Fruit: Tangerine Cnt: 18

10

Fruit: