Jan Chatterjee 23BCI0003

Q1 Similarities

(a) Both classes and structures provide encapsulation of data into a single unit

(b) Member variables are present in both

(c) Structures can hold function pointer like how classes posses member functions

Differences

Structures	Classes
• Cannot hold Null values	• Can hold Null values
• Can be held in stack or heap memory	• Mainly held in heap memory.
• Automatically initialised	• Constructor is made to initialise this
• Does not support inheritance natively	• Inheritance is natively supported
• Structures are held by reference	• Classes are reference types

(Q2) Static objects are those objects that exist until the end of execution. Static objects exist inside the data section of a program, thus it maintains state from function call to function call.

Sometimes static objects are also global. They are declared beyond the scope of any function

These objects are available from anywhere in scope and persist until the program returns from memory.

These objects are declared with keyword 'static'.

(52) Example

```
#include <iostream>
#include <stdlib.h>
#include <vector>

Void func(int ad) {
    Static std::vector<int> lis;
    lis.push_back(ad);
    int n=0;
    for(n=0; n<lis.size(); n++)
        std::cout<<i+1<<" "<<lis[i]<<" ";
    std::cout<<std::endl;
}
int main() {
    Srand(0);
    for(int i=0; i<10; i++)
        func(rand());
}
```

Q3

Reference variables are those variables that contain the address of another variable. Thus we can access a variable by dereferencing an address, thus directly accessing the relevant memory location.

Often times we use reference variables in order to access objects, arrays and other classes.

Reference variables are critical for accessing heap memory, allowing us to use more dynamic programming paradigms and allow us to store and return multiple values.

```
#include <stdio.h>

void changeAddr(int *ref) { (*ref)++; }

void change(int val) { val++; }

int main() {
    int n=9;
    printf("%d\n", n);

    change(n);          // change not carried into main
    printf("%d\n", n);

    changeAddr(&n);     // change is done to n directly.
    printf("%d\n", n);

}
```

Q4

```
#include <corecrt.h>
#include <iostream>
#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
using namespace std;

typedef struct {
  char name[5000];
} chio;

class Stack {
private:
  uint8_t *data;
  size_t len;
  size_t filled;

  inline int BoundCheck(size_t n) { return (len - filled) < n; }

public:
  Stack(size_t ln = 1024) {
    len = ln;
    data = (uint8_t *)malloc(len);
    for (int i = 0; i < len; i++)
      data[i] = 0;
    filled = 0;
  }

  void StackExpand() {
    len *= 1.5;
    data = (uint8_t *)realloc((void *)data, len);
    cout << "Stack Expanded to: " << len << endl;
  }
  void int8Push(int8_t siz) {
    if (BoundCheck(sizeof(int8_t)))
      StackExpand();
    data[filled] = siz;
    filled++;
  }
  template <typename T> void push(T t) {
    if (BoundCheck(sizeof(T))) {
      StackExpand();
      push(t);
      return;
    }

    size_t temp = filled;
    for (filled; filled - temp < sizeof(T); filled++) {
      data[filled] = (uint64_t)t & 0xff;
```

```
      t = (T)((uint64_t)t >> 8);
    }
    int8Push(sizeof(T));
  }
  int8_t int8Pop() {
    filled--;
    return data[filled];
  }

  template <typename T> void pop(T *t) {
    if (filled == 0) {
      cout << "STACK IS EMPTY" << endl;
      return;
    }

    int8_t siz = int8Pop();
    if (siz > sizeof(T)) {
      cout << "POINTER MISMATCH" << endl;
      int8Push(siz);
      return;
    }

    *t = 0;
    size_t temp = filled;
    for (filled; temp - filled < siz; filled--) {
      *t = (T)(((uint64_t)*t << 8) | data[filled - 1]);
    }
    *t = *t;
  }
};

class Student {
private:
  char name[50];
  int Uid;

public:
  Student(int i = 0) {
    name[0] = 0;
    Uid = i;
  }
  void input() {
    cout << "Name? ";
    cin >> name;
  }
  void out() { cout << "Uid: " << Uid << "\tName: " << name << endl; }
};

int main() {
  Stack a = Stack();

  int letteri = 0;
```

```
  cout << "How Many Characters?: ";
  cin >> letteri;
  for (int i = 0; i < letteri; i++) {
    char ti;
    cin >> ti;
    a.push(ti);
  }

  int Numi = 0;
  cout << "How Many Numbers?: ";
  cin >> Numi;
  for (int i = 0; i < Numi; i++) {
    int ti;
    cin >> ti;
    a.push(ti);
  }

  int Obji = 0;
  cout << "How Many Objects?: ";
  cin >> Obji;
  for (int i = 0; i < Obji; i++) {
    Student *ti = new Student(i + 1);
    ti->input();
    a.push(ti);
  }

  // POPIN
  for (int i = 0; i < Obji; i++) {
    Student *ti;
    a.pop(&ti);
    ti->out();
  }

  for (int i = 0; i < Numi; i++) {
    int ti;
    a.pop(&ti);
    cout << ti << endl;
  }

  for (int i = 0; i < letteri; i++) {
    char ti;
    a.pop(&ti);
    cout << ti << endl;
  }

  return 0;
}
```

```
PS C:\Users\catte\OneDrive\Documents\My\Qaround\C++\SoopClass\DA> g++ .\GenerStack.cpp;.\a.exe
How Many Characters?: 3
abc
How Many Numbers?: 2
1 2
How Many Objects?: 3
Name? Jay
Name? Ray
Name? May
Uid: 3  Name: May
Uid: 2  Name: Ray
Uid: 1  Name: Jay
2
1
c
b
a
```

Q5

```
#include <cstddef>
#include <cstdio>
#include <iostream>
#include <locale.h>
#include <stdint.h>
#include <stdlib.h>
#include <time.h>
#include <vector>

int max(int a, int b) {
  if (a > b)
    return a;
  else
    return b;
}

template <typename T> class Vec {
private:
  T *lis;
  size_t at;
  size_t len;

public:
  Vec(int64_t len = 2) {
    at = -1;
    this->len = len;
    lis = (T *)malloc(this->len * sizeof(T));
  }

  void VecExp(int Se = 0) {
    len *= 1.5;
    len = max(Se, len);
    this->lis = (T *)realloc(lis, len * sizeof(T));
    for (int i = at + 1; i < len; i++)
      lis[i] = 0;
  }

  void push(T a) {
    at++;
    if (at >= len)
      VecExp(at + 1);
    lis[at] = a;
  }

  void prin() {
    for (int i = 0; i <= at; ++i) {
      std::cout << lis[i] << " ";
    }
    std::cout << std::endl;
  }
```

```
  T pop() { return lis[at--]; }

  T &operator[](const size_t ind) {
    if (ind >= len)
      VecExp(ind + 1);
    return lis[ind];
  }
  ~Vec() { free(lis); }
};

int absol(int X) {
  if (X > 0)
    return X;
  return -1 * X;
}

#define GUESSLIM 10
int main() {
  srand(time(NULL));

  uint32_t p = rand();
  Vec<int> lis(2);

  int num = 1;

  for (int i = 0; i < GUESSLIM; i += 1) {
    lis.push(num);
    num *= 10;
    num += rand() % 10;
  }
  int gues = 0;
  printf("GUESS [1-%d] ", GUESSLIM);
  std::cin >> gues;
  gues--;
  /* lis.prin(); */
  p *= gues;
  p %= GUESSLIM;
  /* std::cout << p << std::endl; */
  int dis = GUESSLIM - absol(p - gues) - 1;
  printf("You win Rs %d\n", lis[dis]);
  return 0;
}
```

```
PS C:\Users\catte\OneDrive\Documents\My\Qaround\C++\SoopClass\DA> .\a.exe
GUESS [1-10] 4
You win Rs 17336
PS C:\Users\catte\OneDrive\Documents\My\Qaround\C++\SoopClass\DA> .\a.exe
GUESS [1-10] 3
You win Rs 1241767092
PS C:\Users\catte\OneDrive\Documents\My\Qaround\C++\SoopClass\DA>
```