MIS: 612303017

NAME: AMAN BIPIN MORGHADE

ASSIGNMENT: 3 (Stack Operations)

Qs.1)

**twoStacks.h**

```
#ifndef
typedef struct twoStacks {
    int* A;
    int* B;
    int sizeA;
    int sizeB;
    int topA;
    int topB;
} twoStacks;

void init(twoStacks **s);
void push1(twoStacks* s, int x);
void push2(twoStacks* s, int x);
int pop1(twoStacks* s);
int pop2(twoStacks* s);
#endif
```

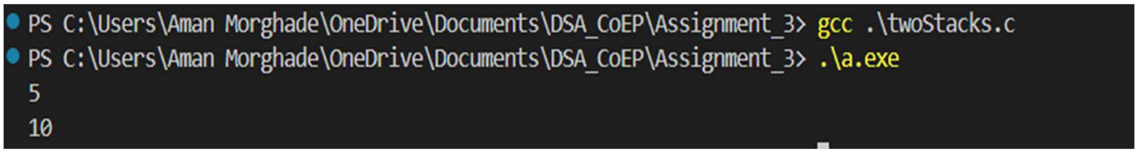
**twoStacks.c**

```
void init(twoStacks** s) {
    *s = (twoStacks*) malloc(sizeof(twoStacks));
    (*s)->A = (int*) malloc(2 * sizeof(int));
    if(!(*s)->A) return;
    (*s)->sizeA = 1;
    (*s)->sizeB = 1;
    (*s)->B = (int*) malloc(2 * sizeof(int));
    (*s)->topA = 0;
    (*s)->topB = 0;
}

void push1(twoStacks* s, int x) {
    s->sizeA++;
    s->A = realloc(s->A, s->sizeA * sizeof(int));
    s->A[s->topA++] = x;
}

int pop1(twoStacks* s) {
    if (s->topA == 0) {;
        return INT_MIN;
    }
    return s->A[--s->topA];
}

void push2(twoStacks* s, int x) {
    s->sizeB++;
    s->B = realloc(s->B, s->sizeB * sizeof(int));
    s->B[s->topB++] = x;
}

int pop2(twoStacks* s) {
    if (s->topB == 0) {
        return INT_MIN;
    }
    return s->B[--s->topB];
}

void freeStacks(twoStacks* s) {
    free(s->A);
    free(s->B);
    free(s);
}
```

**main.c**

```
int main() {
    twoStacks *s;
    init(&s);
    push1(s, 5);
    push2(s, 10);
    printf("%d\n", pop1(s));
    printf("%d\n", pop2(s));
    freeStacks(s);
    return 0;
}
```

**OUTPUT:**



```
PS C:\Users\Aman Morghade\OneDrive\Documents\DSA_CoEP\Assignment_3> gcc .\twoStacks.c
PS C:\Users\Aman Morghade\OneDrive\Documents\DSA_CoEP\Assignment_3> .\a.exe
5
10
```

Qs.2)

**validParenthesis.h**

```
#ifndef
typedef struct Stack {
    char *A;
    int size;
    int top;
} Stack;

void init(Stack *s, int size);
```

```
int isFull(Stack* s);
int isEmpty(Stack *s);
void push(Stack *s, char element);
char pop(Stack* s);
char peek(Stack *s)
#endif
```

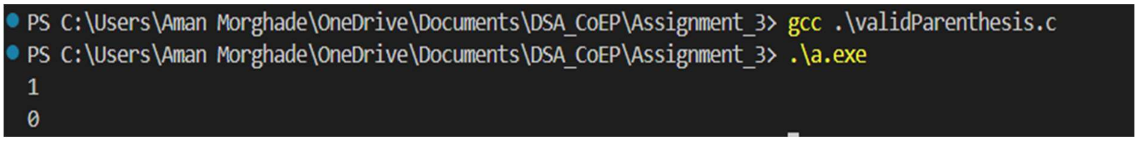
**validParenthesis.c**

```
void init(Stack *s, int size) {
    s->A = (char*) malloc(sizeof(char) * size);
    s->size = size;
    s->top = -1;
    return;
};

int isFull(Stack* s) {
    return s->top == s->size - 1;
}

int isEmpty(Stack *s) {
    return s->top == -1;
}

void push(Stack *s, char element) {
    if(isFull(s)) return;
    s->A[++s->top] = element;
    return;
}

char pop(Stack* s) {
    if(isEmpty(s)) return CHAR_MIN;
    return s->A[s->top--];

}

char peek(Stack *s) {

    if(isEmpty(s)) return CHAR_MIN;

    return s->A[s->top];

}


int isValid(char c, char d) {

    return c == '(' && d == ')' || c == '[' && d == ']' || c == '{' && d == '}';

}


int opening(char c) {

    return c == '(' || c == '[' || c == '{';

}


int closing(char c) {

    return c == ')' || c == ']' || c == '}';

}


int ValidParenthesis(char* array, Stack*s) {

    int len = strlen(array);

    init(s, len);

    for(int i = 0; i < len; i++) {

        if(opening(array[i])) {

            push(s, array[i]);

        } else if(closing(array[i])) {

            if(isValid(peek(s), array[i])) {

                pop(s);

            }

        } else {

            printf("INVALID CHAR");
            return -1;
        }
    }
    return isEmpty(s);
}
```

**main.c**

```
int main() {
    Stack *s;
    printf("%d\n", ValidParenthesis("[()]{}{[()()]()}", s));
    printf("%d\n", ValidParenthesis("[(])", s));
    return 0;
}
```

**OUTPUT**



```
PS C:\Users\Aman Morghade\OneDrive\Documents\DSA_CoEP\Assignment_3> gcc .\validParenthesis.c
PS C:\Users\Aman Morghade\OneDrive\Documents\DSA_CoEP\Assignment_3> .\a.exe
1
0
```

**Qs.3)**

**reverse.h**

```
#ifndef
typedef struct Stack {
    char *A;
    int size;
    int top;
} Stack;

void init(Stack *s, int size);
int isFull(Stack *s);
int isEmpty(Stack *s);
```

```
void push(Stack *s, char element);
char pop(Stack *s);
char peek(Stack *s);
void reverse(Stack *s , char *array);
#endif
```

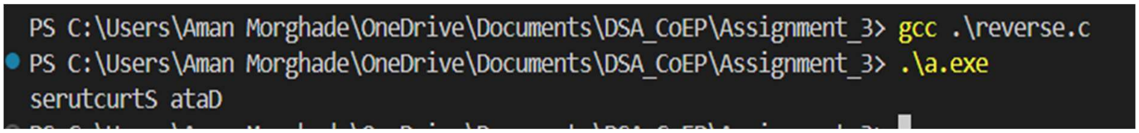
**reverse.c**

```
void init(Stack *s, int size) {
    s->A = (char*) malloc(sizeof(char) * size);
    s->size = size;
    s->top = -1;
    return;
};

int isFull(Stack* s) {
    return s->top == s->size - 1;
}

int isEmpty(Stack *s) {
    return s->top == -1;
}

void push(Stack *s, char element) {
    if(isFull(s)) return;
    s->A[++s->top] = element;
    return;
}

char pop(Stack* s) {
    if(isEmpty(s)) return CHAR_MIN;
    return s->A[s->top--];
}


char peek(Stack *s) {
    if(isEmpty(s)) return CHAR_MIN;
    return s->A[s->top];
}


void reverse(Stack *s , char *array) {
    int len = strlen(array);
    init(s, len);
    for(int i = 0; i < len; i++) {
        push(s, array[i]);
    };
    while(!isEmpty(s)) {
        printf("%c", pop(s));
    };
}
```

**main.c**

```
int main() {
    Stack* s;
    reverse(s, "Data Structures");
    return 0;
}
```

OUTPUT:



```
PS C:\Users\Aman Morghade\OneDrive\Documents\DSA_CoEP\Assignment_3> gcc .\reverse.c
PS C:\Users\Aman Morghade\OneDrive\Documents\DSA_CoEP\Assignment_3> .\a.exe
serutcurtS ataD
```

Qs.4)

**base10_base2.h**

```
#ifndef
typedef struct Stack {
    int *A;
    int size;
    int top;
} Stack;

void init(Stack *s, int size);
int isEmpty(Stack *s);
int isFull(Stack *s);
void push(Stack *s, int element);
int pop(Stack *s);
int peek(Stack *s);
void d2b(Stack* s, int value);
#endif
```

**base10_base2.c**

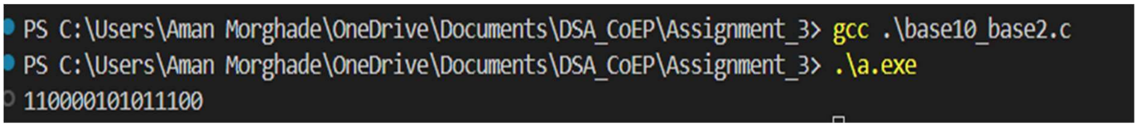
```
void init(Stack *s, int size) {
    s->A = (int*) malloc(sizeof(int) * size);
    s->size = size;
    s->top = -1;
    return;
};

int isFull(Stack* s) {
    return s->top == s->size - 1;
}

int isEmpty(Stack *s) {
    return s->top == -1;
}
```

```
void push(Stack *s, int element) {
    if(isFull(s)) return;
    s->A[++s->top] = element;
    return;
}

int pop(Stack* s) {
    if(isEmpty(s)) return INT_MIN;
    return s->A[s->top--];
}

int peek(Stack *s) {
    if(isEmpty(s)) return INT_MIN;
    return s->A[s->top];
}

void d2b(Stack* s, int value) {
    int rem;
    init(s, 32);
    while(value) {
        rem = value % 2;
        push(s, rem);
        value /= 2;
    }
    while(!isEmpty(s)) {
        printf("%d", pop(s));
    };
    return;
}
```

**main.c**

```
int main() {

    Stack* s;

    d2b(s, 24924);

    return 0;

}
```

OUTPUT:



```
PS C:\Users\Aman Morghade\OneDrive\Documents\DSA_CoEP\Assignment_3> gcc .\base10_base2.c
PS C:\Users\Aman Morghade\OneDrive\Documents\DSA_CoEP\Assignment_3> .\a.exe
110000101011100
```

END