bubble-sort.c

```
#include<stdio.h>
#include<stdbool.h>

#define arraySize 10
int arr[arraySize] = {91, 15, 85, 13, 29, 62, 42, 36,
16, 53};

void header();
void footer();
void arrayContent();
void arrayDuringSorting();

int main()
{
        header();

        printf("\n");
        printf("\n\t| Array Content [BEFORE
SORTING]: \n\n\t\t");
        arrayContent();
        arrayDuringSorting(); // Sorting method
        printf("\n\t| Array Content [AFTER
SORTING]: \n\n\t\t");
        arrayContent();

        footer();
        printf("\n");
        return 0;
}

void arrayDuringSorting()
{
        int temp;
        bool swapped = false;
        printf("\n\t| Array Content [DURING
SORTING]: \n");
        for(int i=0; i<(arraySize-1); i++)
        {
                printf("\n\t * Iteration (%d):
",(i+1));
                printf("[ ");
                for(int a=0; a<arraySize; a++)
                  printf("%d ",arr[a]);
                printf("]\n");
                swapped = false;
                for(int j=0; j<((arraySize-1)-i);
j++)
                {
                        printf("\t    Items
compared: [ %d, %d ] ", arr[j], arr[j+1]);
                        if(arr[j]>arr[j+1])
                        {
                                temp = arr[j];
                                arr[j] =
arr[j+1];
                                arr[j+1] =
temp;

                                swapped =
true;
                                printf("=>
swapped (%d, %d)\n",arr[j], arr[j+1]);
                        }
                        else
                                printf("=> not
swapped\n");
                }
                if(!swapped)
        break;
        }
}

void arrayContent()
{
        for(int i=0; i<arraySize; i++)
        {
                printf("%d ", arr[i]);
        }
        printf("\n");
}

void header()
{
        printf("\n
=====================================
============");
        printf("\n\t\tData Structures and
Algorithm");
        printf("\n\tLesson: Sorting");
        printf("\t\tTitle: Bubble Sort");
        printf("\n    -------------------------------------
------------");
}

void footer()
{
        printf("\n    -------------------------------------
------------");
        printf("\n\t\t    ~ Royland V. Pepaño ~");
        printf("\n\t\t   A 2nd Year BSIT student");
```

```
	printf("\n
================================================
=============\n");
}
```

insertion-sort.c

```
#include<stdio.h>

#define arraySize 10
int arr[arraySize] = {91, 15, 85, 13, 29, 62, 42, 36,
16, 53};

void header();
void footer();
void arrayContent();
void arrayDuringSorting();

int main()
{
	header();

	printf("\n");
	printf("\n\t| Array Content [BEFORE
SORTING]: \n\n\t\t");
	arrayContent();
	arrayDuringSorting(); // Sorting method
	printf("\n\t| Array Content [AFTER
SORTING]: \n\n\t\t");
	arrayContent();

	footer();
	printf("\n");
	return 0;
}

void arrayDuringSorting()
{
	int insert;
	int pos;
	printf("\n\t| Array Content [DURING
SORTING]: \n");
	for(int i=1; i<arraySize; i++)
	{
		printf("\n\t * Iteration (%d): ", i);
		printf("[ ");
		for(int a=0; a<arraySize; a++)
		  printf("%d ",arr[a]);
		printf("]\n");
		insert = arr[i];
		pos = i;
		while(pos > 0 && arr[pos-
1]>insert)
		{
			arr[pos] = arr[pos-1];
			pos--;
			printf("\t\t* %d was
moved to arr[%d].\n", arr[pos], i);
		}
		if(pos!=i)
		{
			printf("\t\t* %d was
inserted at arr[%d].\n", insert, pos);
			arr[pos] = insert;
		}
	}
}

void arrayContent()
{
	for(int i=0; i<arraySize; i++)
	{
		printf("%d ", arr[i]);
	}
	printf("\n");
}

void header()
{
	printf("\n
================================================
=============");
	printf("\n\t\tData Structures and
Algorithm");
	printf("\n\tLesson: Sorting");
	printf("\t\tTitle: Insertion Sort");
	printf("\n     -------------------------------------
------------");
}

void footer()
{
	printf("\n     -------------------------------------
------------");
	printf("\n\t\t    ~ Royland V. Pepaño ~");
	printf("\n\t\t   A 2nd Year BSIT student");
	printf("\n
================================================
=============\n");
}
```

merge-sort.c

```
#include<stdio.h>
```

```
#define arraySize 10

void header();
void footer();
void arrayContent();
void arraySort(int[], int, int);
void arrayMerge(int[], int, int, int);

int main()
{
	int arr[] = {91, 15, 85, 13, 29, 62, 42, 36,
16, 53};
	header();

	printf("\n");
	printf("\n\t| Array Content [BEFORE
SORTING]: \n\n\t\t");
	arrayContent(arr);
	arraySort(arr, 0, arraySize-1);
	printf("\n\t| Array Content [AFTER
SORTING]: \n\n\t\t");
	arrayContent(arr);

	footer();
	printf("\n");
	return 0;
}

void arraySort(int arr[], int start, int end)
{
	if(start<end)
	{
		int mid = (start + end)/2;
		arraySort(arr, start, mid);
		arraySort(arr, mid+1, end);
		arrayMerge(arr, start, mid,
end);
	}
}

void arrayMerge(int arr[], int start, int mid, int end)
{
	int i = start;
	int j = mid+1;
	int k, index = start;
	int temp[arraySize];

	while(i<=mid && j<=end)
	{
		if(arr[i]<arr[j])
		{
			temp[index] = arr[i];
			i+=1;
		}
		else
		{
			temp[index] = arr[j];
			j+=1;
		}
		index++;
	}

	if(i>mid)
	{
		while(j<=end)
		{
			temp[index] = arr[j];
			index++;
			j++;
		}
	}
	else
	{
		while(i<=mid)
		{
			temp[index] = arr[i];
			index++;
			i++;
		}
	}
	k = start;
	while(k<index)
	{
		arr[k] = temp[k];
		k++;
	}
}

void arrayContent(int arr[])
{
	for(int i=0; i<arraySize; i++)
	{
		printf("%d ", arr[i]);
	}
	printf("\n");
}

void header()
{
	printf("\n
=====================================
=============");
	printf("\n\t\tData Structures and
Algorithm");
```

```
	printf("\n\tLesson: Sorting");
	printf("\t\tTitle: Merge Sort");
	printf("\n     -------------------------------------
------------");
}

void footer()
{
	printf("\n     -------------------------------------
------------");
	printf("\n\t\t    ~ Royland V. Pepaño ~");
	printf("\n\t\t   A 2nd Year BSIT student");
	printf("\n
=====================================
=============\n");
}
```

queue.c

```
#include<stdio.h>
#include<stdlib.h>
#include<stdbool.h>

#define size 10

void header();
void footer();
void insert();
void removed();
void display();

int choice, item;
int rear = 0;
int front = 0;
int queue[size];

int main()
{
	header();
	printf("\n");

	bool exit = true;
	while(exit)
	{
		printf("\n\t\t\t\t~ Queue Menu ~");
		printf("\n\n\t\t\t1. Insert");
		printf("\n\t\t\t2. Remove");
		printf("\n\t\t\t3. Display");
		printf("\n\t\t\t4. Exit");
		printf("\n\n\t   | Enter your
choice: ");
		scanf("%d", &choice);
		switch(choice)
		{
			case 1:
				insert();
				break;
			case 2:
				removed();
				break;
			case 3:
				display();
				break;
			case 4:
				exit = false;
				break;
			default:
				printf("\n\t   |
ERROR: Invalid keyword.\n");
		}
		if(exit==true)
			printf("\n
=====================================
=========\n");
		else
			footer();
	}
	printf("\n");
	return 0;
}

void insert()
{
	if(rear==size)
		printf("\n\t   | WARNING:
Queue reached its maximum capacity.\n");
	else
	{
		printf("\t   | Enter a number to
insert: ");
		scanf("%d", &item);
		printf("\t   | Position: %d,
Inserted Value: %d\n", rear, item);
		queue[rear++] = item;
	}
}

void removed()
{
	if(front==rear)
		printf("\n\t   | WARNING:
Queue is empty.\n");
	else
	{
		printf("\t   | Position: %d,
Removed Value: %d\n", front, queue[front]);
```

```
                front++;
        }
}

void display()
{
        if(front==rear)
                printf("\n\t   | WARNING:
Queue is empty.\n");
        else
        {
                printf("\t   | Queue Size:
%d\n\t\t  ", rear);
                for(int i=front; i<rear; i++)
                        printf("\n\t   | Position:
%d, Value: %d", i, queue[i]);
                printf("\n");
        }
}

void header()
{
        printf("\n
===================================
=============");
        printf("\n\t\tData Structures and
Algorithm");
        printf("\n\tLesson: Stack & Queue");
        printf("\t   Title: Queue");
        printf("\n     -------------------------------------
------------");
}

void footer()
{
        printf("\n     -------------------------------------
------------");
        printf("\n\t\t    ~ Royland V. Pepaño ~");
        printf("\n\t\t   A 2nd Year BSIT student");
        printf("\n
===================================
=============\n");
}
```

quick-sort.c

```
#include<stdio.h>

#define arraySize 10

void header();
void footer();
void arrayContent();
void arraySort(int[], int, int);

int main()
{
        int arr[] = {91, 15, 85, 13, 29, 62, 42, 36,
16, 53};
        header();

        printf("\n");
        printf("\n\t| Array Content [BEFORE
SORTING]: \n\n\t\t");
        arrayContent(arr);
        arraySort(arr, 0, arraySize-1);
        printf("\n\t| Array Content [AFTER
SORTING]: \n\n\t\t");
        arrayContent(arr);

        footer();
        printf("\n");
        return 0;
}

void arraySort(int arr[], int start, int end)
{
        int index = start;
        int i, temp;
        int pivot = arr[end];
        if(start<end)
        {
                for(i=start; i<end; i++)
                {
                        if(arr[i]<=pivot)
                        {
                                temp = arr[i];
                                arr[i] =
arr[index];
                                arr[index] =
temp;
                                index++;
                        }
                }
                temp = arr[index];
                arr[index] = arr[end];
                arr[end] = temp;
                arraySort(arr, start, index-1);
                arraySort(arr, index+1, end);
        }
}

void arrayContent(int arr[])
{
        for(int i=0; i<arraySize; i++)
        {
                printf("%d ", arr[i]);
        }
        printf("\n");
}

void header()
{
        printf("\n
=================================================");
        printf("\n\t\tData Structures and
Algorithm");
        printf("\n\tLesson: Sorting");
        printf("\t\tTitle: Quick Sort");
        printf("\n    -------------------------------------
------------");
}

void footer()
{
        printf("\n    -------------------------------------
------------");
        printf("\n\t\t    ~ Royland V. Pepaño ~");
        printf("\n\t\t   A 2nd Year BSIT student");
        printf("\n
=================================================\n");
}
```

selection-sort.c

```
#include<stdio.h>

#define arraySize 10
int arr[arraySize] = {91, 15, 85, 13, 29, 62, 42, 36,
16, 53};

void header();
void footer();
void arrayContent();
void arrayDuringSorting();

int main()
{
        header();

        printf("\n");
        printf("\n\t| Array Content [BEFORE
SORTING]: \n\n\t\t");
        arrayContent();
        arrayDuringSorting(); // Sorting method
        printf("\n\t| Array Content [AFTER
SORTING]: \n\n\t\t");
        arrayContent();

        footer();
        printf("\n");
        return 0;
}

void arrayDuringSorting()
{
        int min;
        printf("\n\t| Array Content [DURING
SORTING]: \n");
        for(int i=0; i<(arraySize-1); i++)
        {
                printf("\n\t * Iteration (%d): ",
i+1);
                printf("[ ");
                for(int a=0; a<arraySize; a++)
                  printf("%d ",arr[a]);
                printf("]\n");

                min = i;
                for(int j=i+1; j<arraySize; j++)
                {
                        if(arr[j]<arr[min])
                                min = j;

                }
                if(min!=i)
                {
                        printf("\t\tItems
Swapped: [%d, %d]\n", arr[i], arr[min]);
                        int temp = arr[min];
                        arr[min] = arr[i];
                        arr[i] = temp;
                }
        }
}

void arrayContent()
{
        for(int i=0; i<arraySize; i++)
        {
                printf("%d ", arr[i]);
        }
        printf("\n");
}

void header()
{
	printf("\n
=====================================
=============");
	printf("\n\t\tData Structures and
Algorithm");
	printf("\n\tLesson: Sorting");
	printf("\t\tTitle: Selection Sort");
	printf("\n	-------------------------------------
------------");
}

void footer()
{
	printf("\n	-------------------------------------
------------");
	printf("\n\t\t	~ Royland V. Pepaño ~");
	printf("\n\t\t	A 2nd Year BSIT student");
	printf("\n
=====================================
=============\n");
}
```

shell-sort.c

```
#include<stdio.h>

#define arraySize 10
int arr[arraySize] = {91, 15, 85, 13, 29, 62, 42, 36,
16, 53};

void header();
void footer();
void arrayContent();
void arrayDuringSorting();

int main()
{
	header();

	printf("\n");
	printf("\n\t| Array Content [BEFORE
SORTING]: \n\n\t\t");
	arrayContent();
	arrayDuringSorting(); // Sorting method
	printf("\n\t| Array Content [AFTER
SORTING]: \n\n\t\t");
	arrayContent();

	footer();
	printf("\n");
	return 0;
}

void arrayDuringSorting()
{
	int inner, outer;
	int insert;
	int interval = 1;
	int elements = arraySize;
	int i = 0;
	printf("\n\t| Array Content [DURING
SORTING]: \n");
	while(interval<=(elements/3))
		interval = interval * 3 + 1;
	while(interval>0)
	{
		printf("\n\t * Iteration (%d): ",
i+1);
		printf("[ ");
		for(int a=0; a<arraySize; a++)
		  printf("%d ",arr[a]);
		printf("]\n");

		for(outer=interval;
outer<elements; outer++)
		{
			insert = arr[outer];
			inner = outer;
			while(inner>(interval-
1) && arr[inner-interval] >= insert)
			{
				arr[inner] =
arr[inner-interval];
				inner -=
interval;
				printf("\t\t*
%d was moved.\n", arr[inner]);
			}
			arr[inner] = insert;
			printf("\t\t* %d was
inserted at arr[%d].\n", insert, inner);
		}
		interval = (interval-1)/3;
		i++;
	}
}

void arrayContent()
{
	for(int i=0; i<arraySize; i++)
	{
		printf("%d ", arr[i]);
	}
	printf("\n");
}
```

```
void header()
{
	printf("\n
=====================================
=============");
	printf("\n\t\tData Structures and
Algorithm");
	printf("\n\tLesson: Sorting");
	printf("\t\tTitle: Shell Sort");
	printf("\n	-------------------------------------
------------");
}

void footer()
{
	printf("\n	-------------------------------------
------------");
	printf("\n\t\t    ~ Royland V. Pepaño ~");
	printf("\n\t\t   A 2nd Year BSIT student");
	printf("\n
=====================================
=============\n");
}
```