/

Comp --- 120

Case Constant:
```
#define CARDS 51
```

Arrays:

- `int list[MAXSIZE] = {1, 2, 3, 4};` ALLOWED
- `int list[MAXSIZE]`
  `list = {1, 2, 3, 4}` Not ALLOWED

C Libraries:

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

Srand((unsigned) time(NULL));
```

Files

```
FILE *inFile;
FILE *outFile;

inFile = fopen("assignment 2.txt", "r");
out File = fopen("data.txt", "w");

fscanf (inFile, "%d", &num);
fprintf(outFile, "%d", num);

fclose(inFile);
fclose (out File);
return 0;

/* Error handling */

 if (inFile == NULL)
 {
   printf("Error opening file");
 }
 if (outFile == NULL)
 {
  printf("Error in outfile");
 }
```

Loops

```
while (i < SIZE)
{
  /* Execute program */
}

Do
{
  /* statements */
} while (i < SIZE);

for(i = 0; i < SIZE; i++)
{
  /* Statements */
}
```

Pseudocode

function: get_valid_denomina

```
1. Read denominator
2. While denominator is 0
   2.1 Print error message
   2.2 Read denominator
   end While
3. Return denominator
End get_valid_denominator
```

Functions

```
#include<time.h>
srand((unsigned)time(Null));

int randomInRange(int lower, int upper)
{
 return rand() % (upper-lower+1) + lower;
}
```

Convert char → int:

```
printf("%d", a[i] - 48)
printf("%d", a[i] - '0'
```

```
float ComputeAverage(const int array[], int numValues)
{
  int i;
  int sum=0;
  for(i=0; i<numValues; i++)
  {
    sum = sum + array[i];
  }
  return (float) sum/numValues;
}
```

Function Prototypes

```
void game();
void ShuffleDect( int [
double foo(char[], int, cha
```

```
int FindMax(const int array[], int numEntries)
{
  int index;
  int max = array[0];
 for (index=0; index<numEntries; index++)
  {
   if(array[index]>max)
   {
     max = array[index]
   }
  }
  return max;
}
```

```
int LinearSearch(int array[], int numEntries, int Value)
{
  int i;
  for(i=0; i<numEntries; i++)
  {
   if(array[i] == value)
   {
     return i;
   }

  return -1;
}
Selection Sort (int data [], int size)

t minIndex;
nt i;
t j;
t holder;

or (i=0; i<Size -1; i++)

  minIndex = i;

  for( j=i+1; j<size; j++)
  {
    if(data [j] < data[minIndex])
      minIndex = j;
  }


holder = data[minIndex];
data[minIndex] = data[i];
data[i] = holder;
}
```

```
int Convert to Int(char array [], int SIZE)
{
  int i;
  int number = 0;

  for (i=0; i<SIZE; i++)
  {
   if (array[i] < '0' || array[i] > '9' )
   ~~ret~~ return -1;
   else
     number = number * 10 + array[i] - '0';
  return number;
}
```

```
t define  Max-SIZE  5
 define  TRUE    1
 define  FALSE   0

nt BinarySearch (int array[], int numItems, int item)

int left = 0;
int right = numItems - 1;
int middle;
int found = FALSE;
int location = -1;
while(left <= right && !found)
{
 middle = (left + right) / 2;
 if(item < array[middle])
  right = middle - 1;
else if(item > array[middle])
left = middle + 1;
else
{ found = TRUE;
  location = middle;
}
[illegible]
```