National University of Computer and Emerging Sciences

FAST School of Computing **Spring-2022** **Islamabad Campus**

CS-1002: Programming Fundamentals

Serial No:

Final Exam
Total Time: 3 Hours
Total Marks: 105

Friday, 16th Dec, 2022

Course Instructors

Dr. Aleem, Dr. Akhter, Ms. Ifrah, Mr. Shehreyar

_____ _____

Signature of Invigilator

Student Name Roll No. Section Signature

DO NOT OPEN THE QUESTION BOOK OR START UNTIL INSTRUCTED.

Instructions:

1. Attempt on question paper. Attempt all of them. Read the question carefully, understand the question, and then attempt it.
2. No additional sheet will be provided for rough work. Use last 3 pages for rough work. Work written on rough pages will not be marked.
3. After asked to commence the exam, please verify that you have **15** different printed pages including this title page and rough work. There are a total of **5** questions.
4. Calculator sharing is strictly prohibited.
5. Use permanent ink pens only. Any part done using soft pencil will not be marked and cannot be claimed for rechecking.
6. Add smiley face in front of instruction number 2 to get 5 bonus marks.

	Q-1	Q-2	Q-3	Q-4	Q-5	Total
Marks Obtained						
Total Marks	**65**	**10**	**10**	**10**	**10**	**105**

Question 1 [13*5=65 Marks]

Write the output of the following C++ programs. Please note that there is no syntax and logical error in the programs.

`int Quad(int n)`
`{`
`  return (n*n*n*n);`
`}`
`int main() {`
`  int num=1634;`
`  int res=0;`
`  int remainder;`
`  int n = num;`
`  while(n!=0)`
`  {`
`  remainder = n % 10;`
`  res = res + Quad(remainder);`
`  n = n / 10;`
`  }`
`  cout<<"\n Result:"<<res;`
`  return 0;`
`}`
	Output:
	
	1634

`int main() {`
`  int y = 2;`
`  switch (y)`
`  {`
`  case 0:  y = y + 11;`
`  case 1:  y = y / 2;`
`  case 2:  y = y * 5;`
`  case 3:  y = y + 1;`
`  default: y = y % 3;`
`  }`
`  cout << y << endl;`
`return 0;`
`}`
	Output:
	
	2

```
int main()
{
      int i, j, m, answer;
      m = 0;
      j = 3;
      while (m < 3) {
            for (i = 0; i < j; i++) {
                  answer = i *m;
                  cout << answer;
            }
            m = m + 1;
            cout << endl;
      }
return 0;
}
```

Output:

000
012
024

```
int main()
{
  int num[5]= {1,2,3,4,5};
  int* p;
  p = num;
  *p = 20;
  p = &num[1];
  *(++p) = 30;
  p = num + 4;
  *p = 30;
  p = num;
  *(p + 3) = 40;
  for (int i = 1; i < 5; i++)
     cout << num[i] << "    ";
  return 0;
}
```

Output:

2 30 40 30

```
int main()
{
  int x[10]={0,1,2,3,4,5,6,7,8,9};
  int *ptr1,*ptr2;
  ptr1=x+2;
  ptr2=&x[9];
  cout<<*ptr1 * *ptr2;
  return 0;
}
```

Output:

18

```
int WHAT(int A[], int N){
    int ANS = 0;
    int S = 0;
    int E = N-1;
    for(S = 0, E = N-1; S < E; S++, E--)
        ANS += A[S] - A[E];
    return ANS;
}
int main(){
    int A[] = {1, 2, 3, 4, -5, 1, 3, 2, 1};
    cout<< WHAT(A, 7);
    return 0;
}
```

Output:

7

```
int main()
{
        int *a, *b, *c;
        int x = 800, y = 300;
        a = &x;
        b = &y;
        *a= (*b) - 200;
        cout<<x<<"   "<<*a;
   return 0;
}
```

Output:

100 100

```
int get(int N=0)
{
    int static x = 0;
    return x++;
}
int main()
{
    const int N = 6;
    int nums[] = { 1,2,3,4,5,6 };
    int idx=1;
    while (idx)
    {
        idx = get(get());
        if (idx >= N)
        {
            break;
        }
        cout << nums[idx] << endl;
    }
      return 0;
}
```

Output:

2
4
6

```
int main()
{
    int i, j, var = 'A';
    for (i = 3; i >= 1; i--) {
        for (j = 0; j < i; j++)
        {
            if(((i+var + j))%4==0)
                continue;
            cout<<char (i+var + j);
        }
        cout<<endl;
    }
      return 0;
}
```

Output:

EF
C
B

```
void Sum(int a) {
      cout << a + 100 << endl;
}
void Sum(int a, int b, int c = 10) {
      cout << a + b + c << endl;
}
int main() {
      Sum('A');
      Sum('B', 30);
      Sum(20, 30, 90.5);
      return 0;
}
```

Output:

165
106
140

```
void find(int , int& , int& ,int=4);
int main() {
      int one=1, two=2, three=3;
      find(one, two, three);
     cout <<one<<","<<two<<","<<three<<endl;
      return 0;
}
void find(int a, int& b, int& c, int d) {
      if(d<1)
          return;
      cout<<a<<","<<b<<","<<c<<endl;
      c = a + 2 * b;
      int temp = b;
      b = a;
      a = 2 * temp;
      d%2?find(b,a,c,d-1):find(c,b,a,d-1);
}
```

Output:

1,2,3
5,1,4
5,2,7
9,5,4
1,5,5

```
char c[7][11] = {"PF-Final","PF","Exam","Students","lazy","2022", "programmer"};
char* add(char* ptr){
    return ptr + 11;
}
char* sub(char* ptr){
    return ptr - 11;
}
int main()
{
    char * mystery=c[4];
    cout<<mystery<<endl;
    cout<<sub(mystery)[2]<<endl;
    mystery= sub(mystery);
    cout<<mystery<<endl;
    cout<<sub(mystery) + 1 <<endl;
    cout<<add(add(mystery))+13<<endl;
    cout<<*add(add(mystery))<<endl;
return 0;
}
```

Output:

```
lazy
u
Students
xam
ogrammer
2
```

```
const int s=3;
int* listMystery(int list[][::s]){
    int i = 1,k=0;
    int *n = new int[::s];
    for(int i=0;i<::s;++i)
           n[i]=0;
    while(i < ::s)
    {
        int j = ::s - 1;
        while(j >= i)
        {
            n[k++]=list[j][i] * list[i][j];
            j = j - 1;
        }
        i = i + 1;
    }
    return n;
}
void displayMystery(int * arr){
    cout<<"[ ";
    for(int i=0;i<::s;++i)
        cout<<arr[i]<<(i!=(::s - 1)?" , ":" ");
    cout<<"]"<<endl;
}
int main(){
    int L[][::s] = {{8, 9, 4}, {2, 3, 4}, {7, 6, 1}};
    int *ptr=listMystery(L);
    displayMystery(ptr);
    delete [] ptr;
    return 0;
}
```

Output:

```
[ 24 , 9 , 1 ]
```

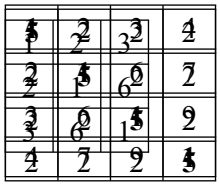
Question 2 [10 Marks]

Complete the C++ code for a function **Mirror-TwoD** which takes a two-dimensional integer array with its rows and column size as input parameters. This function verifies and returns true, if the data elements in array creates a mirror effect, and false otherwise. **Please note that you have to complete the code by writing on the blank lines. Each line should contain only one statement.**

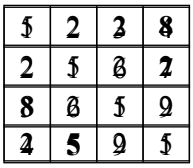
A Mirror array:

1. Has the same number of rows and columns, (Square array)
2. Has the same values on the main diagonal.
3. The data elements and their ordering are same above and below main diagonal.

For Example, the arrays given below are Mirror-TwoD



But the following arrays are **NOT** Mirror-TwoD. (Please look at bold digits)



```
bool Mirror_TwoD(int arr[][4], int rows, int cols) {

     if (rows != cols)

            return ____ false _________;

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

           if (_____arr[0][0] != arr[i][i]_)

                 return false;

            for (int j = i + 1; j < cols; ++j) {

                 if (arr[i][j] != arr[j][i])

                       return false;

                 }

      }

      return true;

}
```

Question 3 [10 Marks]

P_N is the n^{th} Pell number that is generated as follows: $P_N = 2 * P_{N-1} + P_{N-2}$
Write a recursive function `PellNum` to generate N^{th} Pell number for the given value of *N*.
NOTE: 0^{th} Pell number is equal to 0, and 1^{st} P number is equal to 1. Use of any loop, static, or global data-item will result in Zero marks. Please note that you have to complete the code by writing on the blank lines. Each line should contain only one statement.

```
int pellNum(int n)
{
   if(n <= 2)
      return n;
   return 2*pellNum(n-1) + pellNum(n-2);

}

int main() {
   int n;
   cin>>n;
   cout << pellNum(n)<<endl; // Pell number at that position.
   return 0;
}
```

Question 4 [10 Marks]

Complete the following C++ function. The functions finds the middle element of the char array using a **single loop**. For example, if the given char array is "abcde" then function must return character 'c'. If there are even number of characters in the character array, then there would be two middle characters, function should return second middle element. For example, middle character of "abcdef" is 'd'.

Note: more than one single loop, nested loop and built-in functions are not allowed.
Please note that you have to complete the code by writing on the blank lines. Each line should contain only one statement.

```
char* findMidPosition(char * sp)

{

    char* fp=sp;

    while (*fp!= '\0' && *(fp+1)!= '\0' )

    {

     sp=sp+1;
     fp=fp+2;
    }

    return sp;

}
```

Question 5 [10 Marks]

Complete the following function that evaluates a given polynomial. It takes four arguments. The first argument is a pointer to an integer array, which is an array of coefficients ordered from lowest to highest: . The second parameter is an integer size of the array, the third is an integer value x, and the fourth parameter is an integer which is used to return the value of the evaluated polynomial.

Your function should evaluate the polynomial at x:

For example, if the array of coefficients is int arr[] = {2, 3, 1, 2}, size =4, x=4, then the function should return 158 which is obtained by evaluating the polynomial f(x) as shown above.

Your CalcPoly function must follow these rules:

- **You cannot declare any variable inside this function**
- **You cannot change the return type of the function. You should be able to pass the value of the evaluated polynomial back to the main function.**
- **Only while loop is allowed inside the function for evaluation of the polynomial**

```
void CalcPoly(int *ptr, int size, int x, int &value)
{
    while (size-- > 0)
    {
        value += *(ptr + size) * pow(x, size);
    }
}
int main()
{
   int arr[] = { 2, 3, 1, 2 };
   int output=0, x = 4, size=4;
   //Call CalcPoly function properly below
    CalcPoly(arr,size, x, output);
    cout << output
}
```

Rough Work:

Rough Work:

Rough Work: