# Salman Ahmad

# 04072113050

# BSCS 6th Sem


# CS-121 OOP

# Assignment 1

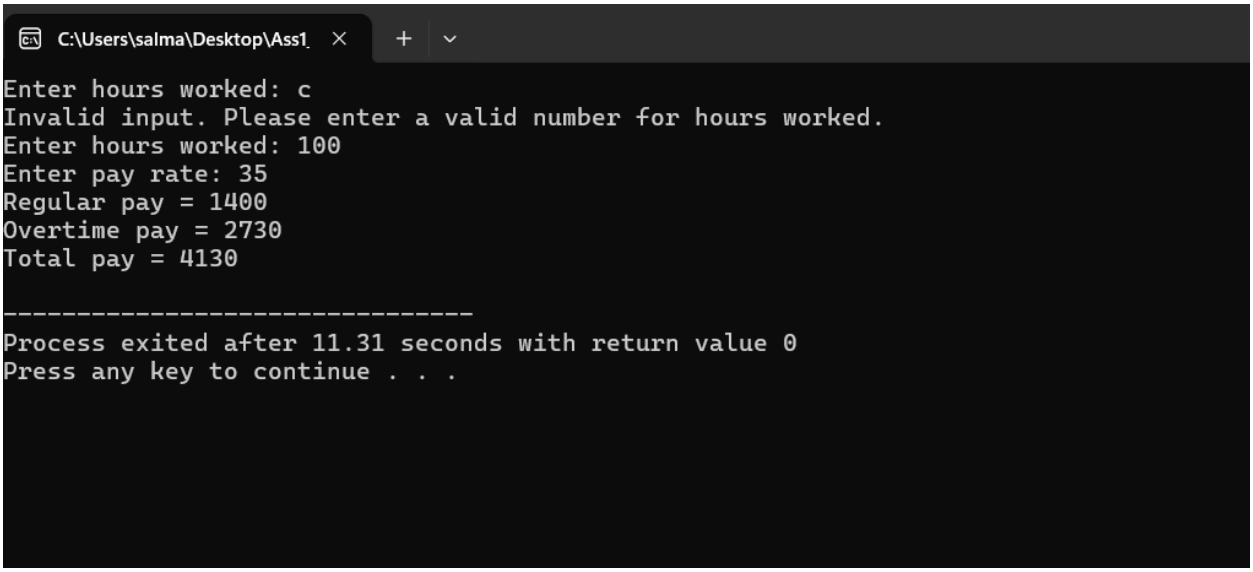
**Q1. Pay Rate program**

```
#include <iostream>
#include <iomanip>

using namespace std;

int main() {
   double hoursWorked, hourlyRate, regularPay, overtimePay, totalPay;
   const double OVERTIME_RATE = 1.30;
   bool validInput = false;

   // Input hours worked with input validation
   while (!validInput) {
```

```
    cout << "Enter hours worked: ";
    if (cin >> hoursWorked && hoursWorked >= 0) {
        validInput = true;
    } else {
        cin.clear();
        cin.ignore(10000, '\n');
        cout << "Invalid input. Please enter a valid number for hours worked." << endl;
    }
}

validInput = false; // Reseting validInput for the next input

// Input pay rate with input validation
while (!validInput) {
    cout << "Enter pay rate: ";
    if (cin >> hourlyRate && hourlyRate >= 0) {
        validInput = true;
    } else {
        cin.clear();
        cin.ignore(10000, '\n');
        cout << "Invalid input. Please enter a valid number for pay rate." << endl;
    }
}

// Computing pay
if (hoursWorked > 40) {
    // regular pay
```

```
    regularPay = 40 * hourlyRate;

    //overtime pay
    double overtimeHours = hoursWorked - 40;
    overtimePay = overtimeHours * hourlyRate * OVERTIME_RATE;
  } else {
    regularPay = hoursWorked * hourlyRate;
    overtimePay = 0.0;
  }

  // Computing total pay
  totalPay = regularPay + overtimePay;

  cout << "Regular pay = " << regularPay << endl;
  cout << "Overtime pay = " << overtimePay << endl;
  cout << "Total pay = " << totalPay << endl;

  return 0;}
```

**output:**

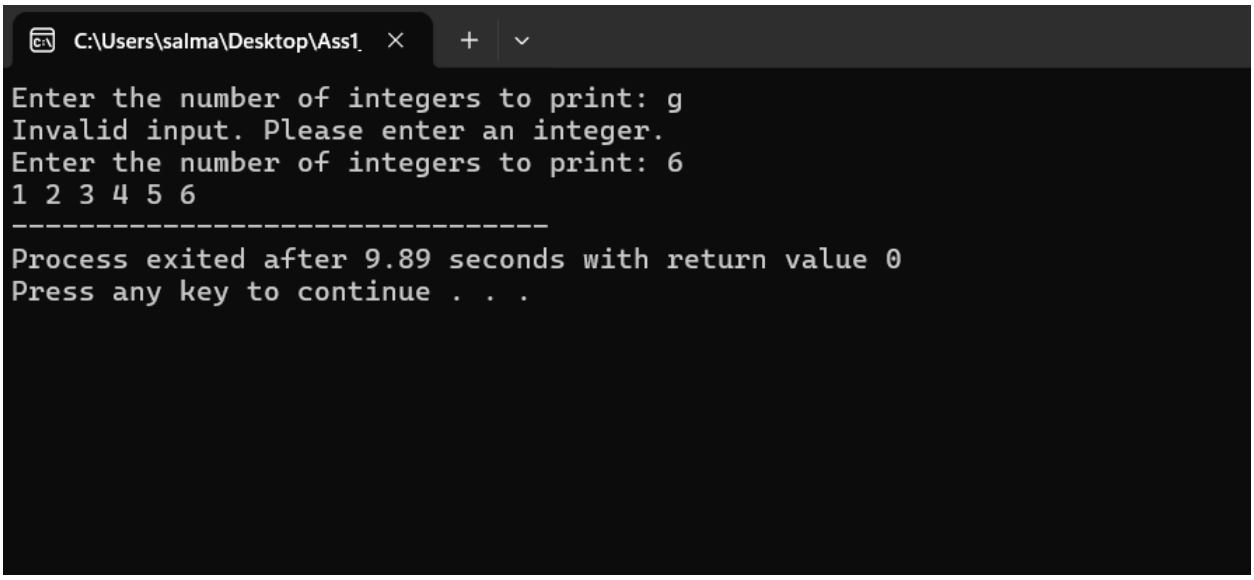


## Q2. Limit From User Program

```
#include <iostream>

using namespace std;

int main() {
  int n;
  bool validInput = false;

//Input Validation for charecter input
  while (!validInput) {
    cout << "Enter the number of integers to print: ";
    if (cin >> n) {
      validInput = true;
    } else {
      // Input case
      cout << "Invalid input. Please enter an integer." << endl;
      cin.clear();
      cin.ignore(10000, '\n');
    }
  }

  // Printing integers
  for (int i = 1; i <= n; ++i) {
    cout << i << " ";
  }
```

```
    return 0;
}
```

**Output:**



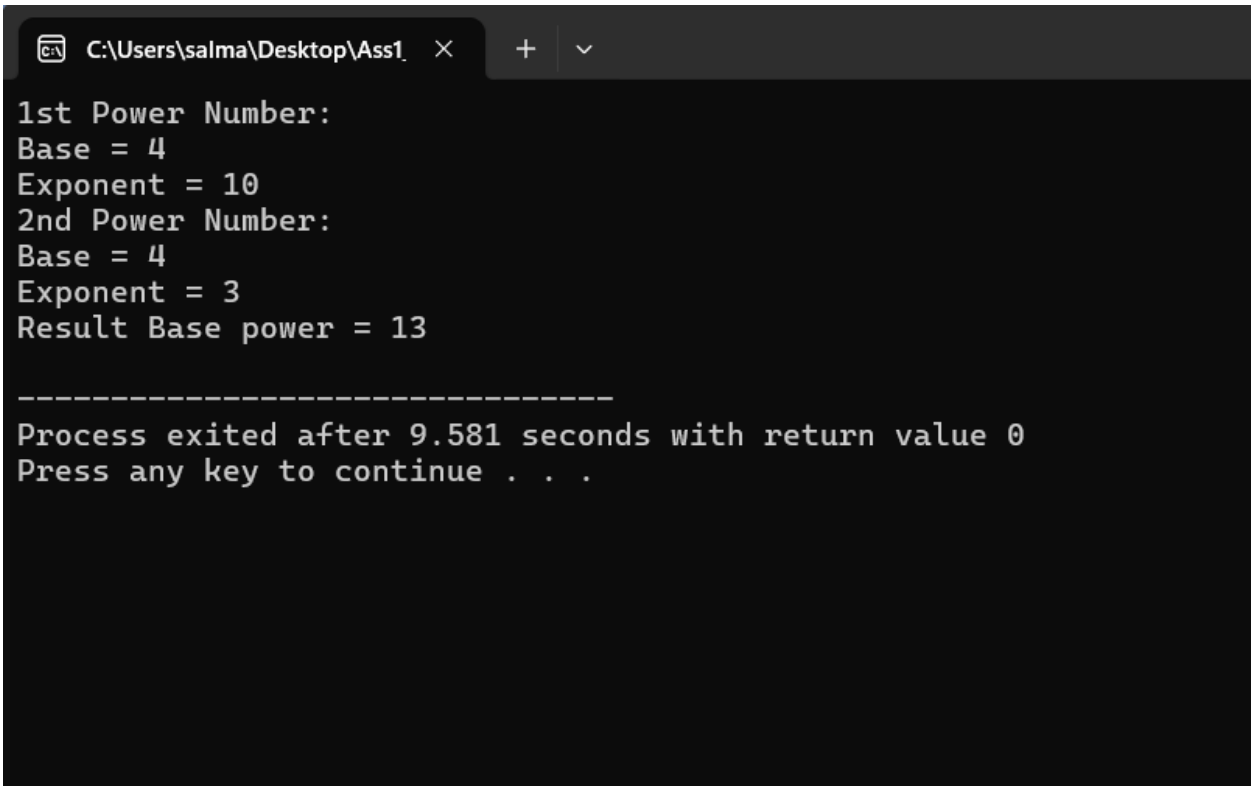

## Q3. Product of power program

```
#include <iostream>

using namespace std;

void getValidatedInput(const string &prompt, int &value) {
    bool validInput = false;
    while (!validInput) {
        cout << prompt;
        if (cin >> value) {
            validInput = true;
```

```
    } else {
      // Invalid input case
      cout << "Invalid input. Please enter a valid integer." << endl;
      cin.clear();
      cin.ignore(10000, '\n');
    }
  }
}

int main() {
  int base1, exponent1, base2, exponent2, result;

  // Input for the first power number
  getValidatedInput("1st Power Number:\nBase = ", base1);
  getValidatedInput("Exponent = ", exponent1);

  // Input for the second power number
  getValidatedInput("2nd Power Number:\nBase = ", base2);
  getValidatedInput("Exponent = ", exponent2);

  if (base1 != base2) {
    cout << "The bases of the power numbers are not the same. Cannot compute the
product." << endl;
    return 1;
  } else {
    result = exponent1 + exponent2;
  }
```

```
    // Display the result
    cout << "Result Base power = " << result << endl;


    return 0;
}
```

**OutPut**:



## Q4. CRUD Operations Program

```
#include <iostream>
#include <string>

using namespace std;

// Course structure
struct Course {
  int course_ID;
  string course_title;
  int credit_hrs;
};

// Function prototypes
void addCourse(Course courses[], int &size);
void updateCourse(Course courses[], int size);
void deleteCourse(Course courses[], int &size);
void displayCourse(const Course courses[], int size);
void displayAllCourses(const Course courses[], int size);

int main() {
  const int MAX_COURSES = 5;
  Course courses[MAX_COURSES];
  int size = 0;
```

```
char choice;
do {
  // Display menu options
  cout << "\nCourse Management Menu:\n";
  cout << "Press 1 to Add a Course\n";
  cout << "Press 2 to Update a Course\n";
  cout << "Press 3 to Delete a Course\n";
  cout << "Press 4 to Search and Display a Course\n";
  cout << "Press 5 to Display All Courses\n";
  cout << "Press e to Exit\n";
  cout << "Enter your choice (1-5) or e: ";
  cin >> choice;
  cin.ignore();


  switch (choice) {
    case '1':
      addCourse(courses, size);
      break;
    case '2':
      updateCourse(courses, size);
      break;
    case '3':
      deleteCourse(courses, size);
      break;
    case '4':
      displayCourse(courses, size);
```

```
            break;
        case '5':
            displayAllCourses(courses, size);
            break;
        case 'e':
            cout << "Exiting program." << endl;
            break;
        default:
            cout << "Invalid choice. Please enter a number between 1 to 5 or e for exit." << endl;
      }
   } while (choice != 'e');

   return 0;
}

// Function to add a course
void addCourse(Course courses[], int &size) {
   if (size >= 5) {
      cout << "Cannot add more courses. Array is full." << endl;
      return;
   }

   Course newCourse;
   cout << "Enter Course ID: ";
   cin >> newCourse.course_ID;
   cin.ignore(); // To ignore newline character
   cout << "Enter Course Title: ";
```

```
    getline(cin, newCourse.course_title);
    cout << "Enter Credit Hours: ";
    cin >> newCourse.credit_hrs;

    courses[size] = newCourse;
    size++;
    cout << "Course added successfully." << endl;
}

// Function to update a course
void updateCourse(Course courses[], int size) {
    int id;
    cout << "Enter Course ID to update: ";
    cin >> id;

    for (int i = 0; i < size; ++i) {
        if (courses[i].course_ID == id) {
            cout << "Enter new Course Title: ";
            cin.ignore(); // To ignore the newline character
            getline(cin, courses[i].course_title);
            cout << "Enter new Credit Hours: ";
            cin >> courses[i].credit_hrs;
            cout << "Course updated successfully." << endl;
            return;
        }
    }
    cout << "Course with ID " << id << " not found." << endl;
```

```
}

// Function to delete a course
void deleteCourse(Course courses[], int &size) {
   int id;
   cout << "Enter Course ID to delete: ";
   cin >> id;

   for (int i = 0; i < size; ++i) {
      if (courses[i].course_ID == id) {
         for (int j = i; j < size - 1; ++j) {
            courses[j] = courses[j + 1];
         }
         size--;
         cout << "Course deleted successfully." << endl;
         return;
      }
   }
   cout << "Course with ID " << id << " not found." << endl;
}

// Function to search and display a course
void displayCourse(const Course courses[], int size) {
   int id;
   cout << "Enter Course ID to search: ";
   cin >> id;
```

```
    for (int i = 0; i < size; ++i) {
        if (courses[i].course_ID == id) {
            cout << "Course ID: " << courses[i].course_ID << endl;
            cout << "Course Title: " << courses[i].course_title << endl;
            cout << "Credit Hours: " << courses[i].credit_hrs << endl;
            return;
        }
    }
    cout << "Course with ID " << id << " not found." << endl;
}

// Function to display all courses
void displayAllCourses(const Course courses[], int size) {
    if (size == 0) {
        cout << "No courses to display." << endl;
        return;
    }

    cout << "List of Courses:\n";
    for (int i = 0; i < size; ++i) {
        cout << "Course ID: " << courses[i].course_ID << endl;
        cout << "Course Title: " << courses[i].course_title << endl;
        cout << "Credit Hours: " << courses[i].credit_hrs << endl;
        cout << "----------------------" << endl;
    }
}
```

**OutPut:**

```
Course Management Menu:
Press 1 to Add a Course
Press 2 to Update a Course
Press 3 to Delete a Course
Press 4 to Search and Display a Course
Press 5 to Display All Courses
Press e to Exit
Enter your choice (1-5) or e: 1
Enter Course ID: 1122
Enter Course Title: OOP
Enter Credit Hours: 4
Course added successfully.

Course Management Menu:
Press 1 to Add a Course
Press 2 to Update a Course
Press 3 to Delete a Course
Press 4 to Search and Display a Course
Press 5 to Display All Courses
Press e to Exit
Enter your choice (1-5) or e: 5
List of Courses:
Course ID: 1122
Course Title: OOP
Credit Hours: 4
----------------------

Course Management Menu:
Press 1 to Add a Course
```

```
Course Management Menu:
Press 1 to Add a Course
Press 2 to Update a Course
Press 3 to Delete a Course
Press 4 to Search and Display a Course
Press 5 to Display All Courses
Press e to Exit
Enter your choice (1-5) or e: e
Exiting program.

--------------------------------
Process exited after 32.92 seconds with return value 0
Press any key to continue . . .
```